DETECTION OF ANOMALOUS AND SUSPICIOUS BEHAVIOR PATTERNS FROM SPATIO-TEMPORAL AGENT TRACES

Boštjan Kaluža

Doctoral Dissertation
Jožef Stefan International Postgraduate School
Ljubljana, Slovenia, May 2013

Supervisor: Prof. Dr. Matjaž Gams, Jožef Stefan Institute, Ljubljana, Slovenia
Co-Supervisor: Dr. Mitja Luštrek, Jožef Stefan Institute, Ljubljana, Slovenia

Evaluation Board:
Prof. Dr. Marko Bohanec, Chairman, Jožef Stefan Institute, Ljubljana, Slovenia
Prof. Dr. Bogdan Filipič, Member, Jožef Stefan Institute, Ljubljana, Slovenia
Prof. Dr. Gal A. Kaminka, Member, Bar Ilan University, Ramat-Gan, Israel

MEDNARODNA PODIPLOMSKA ŠOLA JOŽEFA STEFANA
JOŽEF STEFAN INTERNATIONAL POSTGRADUATE SCHOOL

Boštjan Kaluža

DETECTION OF ANOMALOUS AND SUSPICIOUS BEHAVIOR PATTERNS FROM SPATIO-TEMPORAL AGENT TRACES

Doctoral Dissertation

ODKRIVANJE NENAVADNIH IN SUMLJIVIH VZORCEV OBNAŠANJA IZ PROSTORSKO-ČASOVNIH SLEDI AGENTA

Doktorska disertacija

Supervisor: Prof. Dr. Matjaž Gams

Co-Supervisor: Dr. Mitja Luštrek

Ljubljana, Slovenia, May 2013

Abstract

Many applications, including smart environments, surveillance, human-robot interaction, and ambient assisted living, involve the problem of learning patterns of agent behavior from sensor data. Deviant behavior is a pattern in the data that either does not conform to the expected behavior, that is, anomalous behavior, or matches previously defined unwanted behavior, that is, suspicious behavior. The present thesis focuses on the detection of behavior patterns representing a security risk, health problem, or other abnormal behavior contingency.

Real-life applications for deviant behavior detection present several challenges. First, plan recognition research has assumed that atomic actions are either given or can be trivially obtained, while real-life applications require activity recognition from raw sensor readings. The second challenge is how to flexibly encode complex, unstructured, daily-living behavior patterns that do not follow predefined scenarios. Thirdly, deviant behavior may be reflected on different time scales and different modalities, which raises the question of how to combine different time scales and modalities into a single evaluation. Finally, many domains include behavior in which no single event is sufficient to decide whether the behavior is deviant; therefore, an advanced approach is required to accumulate deviation over time.

This thesis proposes a unified framework to analyze agent behavior from prior knowledge and external observations in order to detect deviant behavior patterns, regardless of whether the observed entities are humans, software agents, or even robots. To address the problem of activity recognition from sensor data, the thesis introduces an activity recognition pipeline that includes filtering, attribute construction, activity identification, and activity smoothing. From the behavior analysis perspective, we propose a novel, efficient encoding that we refer to as a spatio-activity matrix. This matrix is able to capture behavior dynamics in a specific time period using spatio-temporal features, whereas its visualization allows visual comparison of different behavior patterns. The thesis also provides a feature extraction technique, based on principal component analysis, in order to reduce the dimensionality of the spatio-activity matrix. We then introduce a clear problem definition that helps establish a theoretical framework for detecting anomalous and suspicious behavior from agent traces in order to show how to optimally perform detection. We discuss why detection error is often inevitable and prove the lower error bound, and provide several heuristic approaches that either estimate the distributions required to perform detection or to directly rank the behavior signatures using machine learning approaches. The established theoretical framework is extended to show how to perform detection when the agent is observed over longer periods of time and no significant event is sufficient to reach a decision. We specify conditions that any reasonable detector should satisfy, analyze several detectors, and propose a novel approach, referred to as a F-UPR detector, that generalizes utility-based plan recognition with arbitrary utility functions. The unified framework is demonstrated empirically in three studies. The first study addresses detection of decreased behavior that indicates disease or deterioration in the health of elderly persons, while the second study deals with the detection of suspicious passengers in the airport simulation. Finally, the third study concerns the verification of persons at an access control point in high-security application.

Povzetek

Aplikacije na področjih pametnih okolij, video nadzora, interakcije človek-robot in ambientalno podprtega življenja običajno vključujejo problem učenja vzorcev obnašanja agenta iz senzorskih podatkov. Odklonsko obnašanje je vzorec v podatkih, ki se bodisi ne ujema s pričakovanim obnašanjem, kar ustreza nenavadnemu obnašanju, bodisi se ujema s predhodno definiranim nezaželenim obnašanjem, kar ustreza sumljivemu obnašanju. Pričujoča disertacija se osredotoča na detekcijo vzorcev, ki lahko predstavljajo varnostno grožnjo, zdravstveni problem ali kakršnokoli drugo tveganje, povezano z obnašanjem agenta.

Pri aplikacijah v realnem življenju se soočamo s številnimi izzivi. Raziskave na področju razpoznavanja planov so predpostavile, da so osnovne akcije agenta podane ali pa jih je mogoče enostavno pridobiti, medtem ko mnoge aplikacije v realnem življenju zahtevajo prepoznavanje akcij iz surovih senzorskih podatkov. Drugi izziv je kako predstaviti zapleteno, nestrukturirano obnašanje ljudi, ki ne sledijo vnaprej določenim vzorcem. Tretji izziv predstavlja dejstvo, da se odklonsko obnašanje lahko odraža na različnih časovnih intervalih in preko različnih zaznavnih vhodov, pri čemer se poraja vprašanje kako združevati različne časovne intervale in zaznavne vhode pri pridobivanju zanesljive ocene obnašanja. In nenazadnje, v mnogih domenah je prisotno obnašanje, kjer iz posameznega zaznanega dogodka ni mogoče sklepati ali je obnašanje odklonsko ali ne, zato je potrebno vpeljati pristop, ki lahko kopiči ocene obnašanja v daljših časovnih obdobjih.

V pričujoči disertaciji predstavimo enoten okvir za analizo obašanja agenta na podlagi predhodnega znanja in zunanjih opažanj. Namenjen je odkrivanju odklonskega obnašanja agentov, ne glede na to ali je predmet opazovanja človek, programski agent ali robot. Disertacija najprej predstavi cevovod za razpoznavanje aktivnosti, ki vključuje odstranjevanje šuma, izdelavo značilk, identifikacijo aktivnosti in izravnavanje šuma pri razpoznavanju. V nadaljevanju opiše novo predstavitev, poimenovano prostorsko-akcijska matrika, namenjeno analizi obnašanja. Z matriko je mogoče z uporabo prostorsko-akcijskih značilk opisati dinamiko obnašanja v določenem časovnem obdobju ter grafično ponazoriti primerjavo med različnimi vzorci obnašanja. Predstavljen je postopek, ki s pomočjo analize glavnih komponent zmanjša dimenzije matrike ter poda njene značilke. V disertaciji se nato osredotočimo na definicijo problema in vzpostavimo formalni okvir za detekcijo nenavadnega in sumljivega obnašanja. Na podlagi formalnega okvira razložimo, zakaj je napaka pri detekciji običajno neizogibna, podamo dokaz za spodnjo mejo napake in predstavimo številne približne metode, ki bodisi neposredno ocenijo porazdelitve, potrebne za detekcijo, bodisi razvrstijo vzorce obnašanja z uporabo strojnega učenja. Formalni okvir je nato razširjen z možnostjo zaznavanja odklonskega obnašanja v daljšem časovnem obdobju, kjer posamezen dogodek ne zadostuje za odločitev. Disertacija poda pogoje, ki jih mora detektor izpolnjevati, in predstavi nov pristop poimenovan detektor F-UPR, ki posploši razpoznavanje planov na podlagi koristnosti s poljubnimi funkcijami koristnosti. Uporabo enotnega okvira za analizo obnašanja agenta predstavimo v treh empiričnih študijah. Prva študija se nanaša na detekcijo obnašanja, ki nakazuje poslabšanje zdravstvenega stanja starejšega posameznika, medtem ko se druga ukvarja z detekcijo sumljivih potnikov na simuliranem letališkem terminalu. Tretja študija zadeva preverjanje identitete vstopajoče osebe v visoko varovanih kontrolnih točkah vstopa.

Contents

List of Figures

List of Tables

List of Algorithms

Abbreviations

AAL	...	ambient assisted living
ADL	...	activities of daily living
AmI	...	ambient intelligence
ARPipe	...	activity-recognition pipeline
ACC	...	accuracy
BN	...	Bayesian net
C4.5	...	C4.5 algorithm
CHMM	...	coupled hidden Markov model
DLD	...	daily-living dynamics
FFT	...	fast Fourier transform
FN	...	false negative
FP	...	false positive
F-UPR	...	utility-function based plan recognition
GPS	...	global positioning system
HCI	...	human-computer interaction
ID	...	intrusion detection
k-NN	...	k-nearest neighbors
LOF	...	local outlier factor
HMM	...	hidden Markov model
ML	...	machine learning
OWL	...	web ontology language
PCA	...	principal component analysis
RFID	...	radio-frequency identification
ROC	...	receiver operating characteristic (curve)
RTLS	...	real-time location system
SWRL	...	semantic web rule language
TN	...	true negative
TP	...	true positive
SGBC	...	sequential grammar-based classifier
SVM	...	support vector machine
UPR	...	utility-based plan recognition
UWB	...	ultra-wideband

Symbols

$\mathbf{x}_t$	...	observation vector at time step t
$\mathbf{f}_t$	...	feature vector at time step t obtained from $\mathbf{x}_t$
T	...	upper limit for time-series observation in period $1 \leq t \leq T$
$\mathcal{B}_T$	...	feature vector dataset
$\mathbf{W}$	...	overlapping feature window
$\mathbf{X}$	...	sequence of observation vectors
$\mathbb{A} = \{a_i\}$	...	a set of possible activities
$a_{i,j}$	...	an activity in time span $i \leq t \leq j$, $a_{i,j} \in \mathbb{A}$
a_t	...	an activity at time t, $a_t \in \mathbb{A}$, also atomic action
$\mathbf{a}^{(T)}$	...	activity sequence of length T
$\mathbb{B} = \{b_i\}$	...	a set of compound activities, also behaviors
b_k	...	a compound activity describing $\mathbf{a}^{(k)}$
$\mathbb{I} = \{\chi_i\}$	...	a set of interactions
$\chi(\langle \mathbf{a}_A, \mathbf{a}_B \rangle)$	...	an interaction describing activity sequences of agents A and B
$\mathbb{S} = \{s_i\}$	...	a set of static landmarks in the environment
s_i	...	a static landmark, $b_s, a \in \mathbb{S}$
$\mathbf{b}$	...	behavior trace, a sequence of tuples $\langle a, s \rangle_t$
$\mathcal{B}_T$	...	behavior trace dataset
$\mathbf{M}(\mathbf{b})$	...	spatio-activity matrix
$\mathbf{m}$	...	spatio-activity matrix $\mathbf{M}$ unrolled into a vector
$\tilde{\mathbf{b}}$	...	behavior signature
$s(\tilde{\mathbf{b}})$	...	probability that signature is generated by suspicious agent
$n(\tilde{\mathbf{b}})$	...	probability that signature is generated by normal agent
$\mathcal{B}$	...	behavior signature dataset
D_g	...	graded detector
D_b	...	binary detector
e_t	...	trigger event
$\mathbf{e}^{(k)}$	...	trigger event trace
$\mathcal{E}$	...	trigger event dataset
$\Pr$	...	probability of an item
$s(e_t)$	...	probability that event is generated by suspicious agent
$n(e_t)$	...	probability that event is generated by normal agent
$\mathcal{D}$	...	event trace dataset
$\theta(\mathbb{H}, \mathbb{A}, \delta, \nu, \pi)$	...	HMM model
$\mathbb{H} = \{h_i\}$	...	a set of HMM hidden states in HMM
$\delta = \{\delta_{ij}\}$	...	HMM state transition probability distribution
$\nu = \{\nu_j(k)\}$	...	HMM state observation probability distribution

$\pi = \pi_i$ ... HMM initial state distribution

1 Introduction

The problem of learning behavior patterns from sensor data arises in many applications including smart environments, video surveillance, network analysis, human-robot interaction, and ambient assisted living. Our focus is on detecting behavior patterns that deviate from regular behaviors and might represent a security risk, health problem, or any other abnormal behavior contingency. In other words, deviant behavior is a data pattern that either does not conform to the expected behavior (anomalous behavior) or matches previously defined unwanted behavior (suspicious behavior). Deviant behavior patterns are also referred to as outliers, exceptions, peculiarities, surprise, misuse, etc. Such patterns occur relatively infrequently; however, when they do occur, their consequences can be quite dramatic, and often negatively so. Typical examples include credit card fraud detection, cyber-intrusions, and industrial damage.

This thesis targets a large class of problems with complex, spatio-temporal, sequential data generated by an entity capable of physical motion in environment, regardless of whether the observed entity is human, software agent, or even robot. In such domains, an agent often has an observable spatio-temporal structure, defined by the physical positions relative to static landmarks and other agents in environment. We suggest that this structure, along with temporal dependencies and patterns of sequentially executed actions, can be exploited to perform deviant behavior detection on traces of agent activities over time. Examples of such detection include: elderly persons, who are being monitored in their smart home and faces a gradual decrease in his health; a reckless driver zigzagging across two lanes; an attacker that tries to gain access at a high-security access point with a stolen identity; and a potentially suspicious passenger at the airport that appears to turn away in a presence of a police officer, but not blatantly so, hence no single observation is enough to raise a suspicion.

1.1 Problem Formulation

The general problem of deviant behavior detection from an agent's sequential spatio-temporal traces is related to the problem of keyhole plan recognition. We use the term **agent** to denote an intelligent, independent entity capable of physical motion and action, such as humans, simulated entities in virtual environments, or robots (Sukthankar and Sycara, 2008). **Plan recognition** refers to inferring the plan, or plans, of an intelligent agent from action observations in the environment (Schmidt et al., 1978). In **keyhole** plan recognition, the observed agent is unaware of, or indifferent to, being observed, whereas **intended** plan recognition assumes that the agent actively cooperates by choosing actions to make its intentions clear to the observer. By contrast, **obstructed** plan recognition assumes that the agent actively obstructs the plan recognition process (Waern and Stenborg, 1995). Our work follows the assumptions of keyhole plan recognition, but it is not restricted to plan recognition only; instead, behavior is represented by patterns, as defined below.

Agents are observed via **spatio-temporal traces**, a vector time series of the agent's physical positions and other sensor data describing the agent's state, such as inertial information, action, or activity. Such vectors are used to determine **agent behavior**, a term that refers to the agent's responses to various perceptual inputs, whether those responses are overt or covert, and voluntary or involuntary. In other words, behavior is the range of actions and mannerisms made by an intelligent agent in conjunction with its environment, situation, and other agents.

From a complete set of observed spatio-temporal traces, we recognize and identify the following characteristics:

- **Actions and activities**: Actions and activities are defined as behavior primitives; that is, elements that help explain and describe the observed behavior of an agent in a specific time span.
- **Behavior signature**: Agent behavior is presented in the form of **behavior signature**, such as a plan or pattern that encodes agent actions and responses to a situation over a period of time.
- **Degree of deviation**: Behavior signature is compared to reference behavior signatures and expressed as a degree of deviation, which measures the likelihood that the observed behavior does not conform to the desired behavior.

We use the term **deviant behavior** to denote agent behavior that deviates from regular behavior of the same agent or other agents. There are two approaches to deviant behavior detection (Avrahami-Zilberbrand, 2009): *suspicious* and *anomalous* behavior detection. **Suspicious behavior** detection assumes a behavior library that encodes *negative* behavior signatures; that is, patterns are considered unwanted or suspicious as they correspond to an identifying match in the library. **Anomalous behavior** detection uses the behavior library in an inverse fashion, encoding only positive behavior signatures. When an observed behavior cannot be matched against the library, it is considered anomalous.

1.2 Challenges

Deviant behavior detection is related to problems such as novelty detection (Markou and Singh, 2003), rare class mining (Elkan, 2001), chance discovery (Ohsawa, 2009), exception mining (Luo et al., 2008), and black swan events (Taleb, 2007). The common key challenges include defining a representative library of behavior signatures, the availability of labeled data for training/validation, dealing with noisy data, modeling normal behavior that keeps evolving, and different application domains' differing notions of an outlier. The class of problems tackled in this thesis, that is, problems with complex, spatio-temporal sequential data generated by an agent moving in a physical environment, poses several additional challenges.

The first challenge is how to recognize atomic activities that constitute behavior patterns. Previous work in plan recognition assumes that atomic actions are either given or trivially obtained, while real-life applications require recognition from raw, and often multimodal, sensor readings.

The second challenge is how to present complex, real-life behavior patterns that do not follow predefined scenarios. Presentation must be robust and flexible to describe sequential spatio-temporal traces compactly.

Third, deviant behavior may reflect on (i) different time scales, and (ii) different modalities. For example, an elderly person can quickly start limping after a minor stroke, which can be detected within hours with accelerometers attached to ankles, or can slowly start limping due to arthritis, which can be detected by comparing month-to-month behavior of daily activities (since the change is not significant for hourly comparison). The question is how to combine different time scales and modalities into a single evaluation.

Finally, many domains include behavior where no single event is sufficient to decide whether behavior is deviant or not. There are three issues that need to be addressed. First, there is no single significant event or incident that would help to immediately reach a decision; rather the observed sequence is a series of observations that allow a decision. Second, there is no knowledge about the exact plans devised by the observed agent. Third, the behavior pattern's deviance degree depends on the past agent behavior. For example, a subsequent deviant pattern is evaluated differently than the first one, since the prior behavior indicates a tendency for deviant behavior. Hence, the simple counting of deviant patterns cannot be applied, since it accumulates all observations linearly. Furthermore, most of the plan recognition methods, which rely on a plan library, are insufficient, since plans are not known in advance. Hence, an advanced approach is required to combine and accumulate deviation over time.

1.3 Approach and Hypothesis

There are four general evidence classes that are potentially valuable for deviant behavior detection:

1. spatio-temporal relationship of agent movement between landmarks fixed over a period of time,
2. temporal dependencies between atomic actions in behavior patterns,
3. time scales and modalities at which behavior patterns are processed, and
4. behavior patterns that can be considered deviant when repeated.

The hypothesis is that it is possible to leverage the available spatio-temporal cues, temporal dependencies, various time scales and modalities, and repetitive behavior patterns to detect anomalous and suspicious behavior.

Spatio-temporal relationships and temporal dependencies: Unlike the existing methodology, which tries to recognize exact or flexible behavior patterns or describe them, our proposed method focuses on activity dynamics and explores the relations between the spatial information and the activities. Spatio-temporal cues assume that the positive behavior patterns of the observed entity can be learned over time since they remain stable.

Time scales and modalities at which behavior patterns are processed: Most of the related work focuses on one specific viewpoint, be it in terms of time scale or sensor modality. Our main idea is to consider various aspects and hypotheses about a behavior pattern and the environment in order to construct a situational awareness and then, on this basis, make a reliable deviation estimation.

Repetitive behavior patterns: The main question addressed is how to combine multiple events to decide whether an event trace corresponds to normal or a deviant agent behavior.

1.4 Scientific Contributions

This thesis led to the following original contributions:

1. A unified anomalous and suspicious behavior detection framework, incorporating the elements below, as well as demonstration on real-world domains.
2. Problem definition and theoretical analysis of anomalous and suspicious behavior detection from agent traces, including optimality conditions and error bounds.
3. New heuristic functions for detecting deviant agent behavior observed over longer periods of time where no significant event is sufficient to reach a decision.
4. New representation of spatio-temporal behavior patterns that allows visual comparison of various patterns and can be efficiently deployed in anomaly detection algorithms.
5. A comprehensive and flexible approach to activity recognition that addresses sensor noise and activity mislabeling to provide activity primitives at various abstraction levels (that is, atomic activities and compound activities).

1.5 Overview of the Thesis Structure

This thesis comprises 11 chapters, organized in two parts as shown in Figure 1.1. Chapter 2 presents the background and surveys the related activity recognition work and anomalous and suspicious behavior detection.

Chapter 1: Introduction
Chapter 2: Related Work

Part I: **Deviant Behavior Detection**
Chapter 3: Activity Recognition
Chapter 4: Behavior Signatures
Chapter 5: Anomalous and Suspicious Behavior Detection
Chapter 6: Accumulating Behavior Evaluations Over Time
Chapter 7: A Unified Detection Framework

Part II: **Empirical Studies**
Chapter 8: Ambient Assisted Living Domain
Chapter 9: Surveillance Domain
Chapter 10: Security Domain

Chapter 11: Conclusions

Figure 1.1: Thesis consists of 11 chapters structured in two parts.

Chapters 3–7 constitute Part I of the thesis, which gradually introduces components of the unified detection framework. Chapter 3 deals with activity recognition and introduces activity recognition pipeline as well as compound activity recognition and the recognition of agent-agent interactions. Chapter 4 then presents the spatio-activity matrix approach to encode daily-living behavior patterns along with a visualization technique and a dimensionality reduction approach. Next, Chapter 5 establishes a formal detection framework, theoretically analyzes detection optimality and error bounds, and proposes several heuristics. Chapter 6 then further extends the framework to address the problem of repeated detection and proposes the F-UPR approach to accumulating suspicion over time. Finally, Chapter 7 connects all the components into a unified detection framework.

Chapters 8–10 constitute Part II of the thesis, which demonstrates how the framework is applied in three real-world domains. First, Chapter 8 focuses on the ambient assisted living domain, where the goal is to assess an elderly person's well-being to detect anomalies in daily-living patterns. Second, Chapter 9 targets a class of applications where no single event is sufficient to determine whether behavior of an agent is suspicious or not; that is, suspicious passenger detection at an airport and dangerous driver detection. Third, in Chapter 10, the unified framework is utilized to improve security at a biometric access point using several modalities.

Finally, Chapter 11 summarizes the thesis, outlines the main contributions and discusses future work.

1.6 Publications

A number of previous publications underlie this thesis. The initial work on activity recognition was published by Luštrek and Kaluža (2009). To address the challenges caused by sensor noise, Kaluža and Dovgan (2009) developed and published pre-processing filtering techniques, while Kaluža (2009) published the removal of spurious activity transitions (post-processing). The complete activity recognition pipeline was then fully applied and first published at the *European Conference on Ambient Intelligence* (Luštrek et al., 2009). This publication enabled analyzing high-level behavior patterns such as spatio-activity matrices, which in turn was published at the *International Conference on Machine Learning and Data Mining* (Kaluža and Gams, 2010) and won the best student paper award. The paper was then further extended and published in *Journal of Ambient Intelligence and Smart Enviroments* (Kaluža and Gams, 2012).

The initial ideas for repeated anomalous and suspicious behavior detection's theoretical foundations were published at the PAIR workshop at the *AAAI Conference on Artificial Intelligence* (Kaluža et al., 2011e), and then distilled along with the F-UPR detector as a full paper at the *International Conference on Autonomous Agents and Multiagent Systems* (Kaluža et al., 2012b).

Empirical studies on the ambient assisted living domain were also published at the *International Joint Conference on Ambient Intelligence* (Kaluža et al., 2010b) and demonstrated at the *European Conference on Artificial Intelligence* (Luštrek et al., 2012) and the *International Conference on Autonomous Agents and Multiagent Systems* (Kaluža et al., 2012a). Results on the security domain were published in the *Journal of Ambient Intelligence and Smart Environments* (Dovgan et al., 2010b) and *Expert Systems with Applications* (Kaluža et al., 2011c). The comprehensive list of related publications is collected in Appendix B.

2 Related Work

In this chapter, we review the related work in two research areas: activity recognition, which includes activity recognition in computer vision and sensor-based activity recognition; and anomalous and suspicious behavior detection based on pattern analysis, transaction analysis, and plan recognition.

2.1 Activity Recognition

Activity recognition is the process whereby an agent's behavior and its situated environment are monitored and analyzed to infer the undergoing activities (Chen et al., 2012). Researchers from different application domains have investigated activity recognition for the past decade by developing a diversity of approaches. We broadly classify activity recognition in categories based on monitoring facilities, which are responsible for capturing contextual information for activity recognition systems to infer agent's activity. There are currently two main activity recognition approaches: vision-based and sensor-based activity recognition.

2.1.1 Vision-Based Activity Recognition

Tracking and understanding the behavior of agents through videos has been a research focus for a long period due to its important role in areas, such as human-computer interaction and surveillance. In vision-based activity recognition, researchers have attempted a wide variety of methods, such as optical flow, Kalman filtering, hidden Markov models, and conditional random fields, under different modalities such as single camera, stereo, and infrared. In addition, researchers have considered multiple aspects on this topic, including single agent tracking, multiple-agent tracking, activity recognition, compound-activity recognition, and finally recognition of multi-agent interactions.

The activity recognition process is typical composed of four steps, namely agent detection, agent tracking, activity recognition and then a high-level activity evaluation. Chen and Khalil (2011) in their review conclude that while significant progress has been made, vision-based activity recognition approaches suffer from issues related to scalability and reusability due to complexity of real world settings; that is, high variability of activities and environment. In addition, cameras are in some communities perceived as invasive, which may prevent this approach from large-scale uptake in some applications, such as home environments.

2.1.2 Sensor-Based Activity Recognition

Sensor-based activity recognition exploits a wide range of sensors, including accelerometers, RFID tags, audio and motion detectors, to name but a few, to monitor an agent behavior along with its environment. These sensors differ in purpose, technical infrastructure, output signals, and underpinning theoretical principles. However, they can be classified in two main categories in terms they are deployed in activity monitoring applications (Chen et al., 2012): wearable sensors and embedded sensors.

Wearable Sensors

Wearable sensors are positioned directly or indirectly on the body of an agent to generate signals while the agent performs activities. When the observed entity is human, wearable sensors can be embedded into clothes, eyeglasses, waists, shoes, mobile device, or positioned directly on the body. They can be used to collect information, such as position, velocity and acceleration of various body parts, pulse, and skin temperature. In the following, we summarize the inertial sensors (for example, accelerometers, gyroscopes, magnetometers), vital sign sensors (heart rate, temperature), and visual markers.

Accelerometer sensors are probably the most frequently exploited wearable sensors, since they are both inexpensive and effective. The first generation of methods was based on a tri-axial accelerometer with threshold algorithms (Kangas et al., 2008). Bourke and Lyons (2008) introduced a threshold algorithm to distinguish between normal activities (sitting down and standing up, lying down and standing up, getting in and out of a car seat, walking etc.) and falls. The ability to discriminate was achieved using a bi-axial gyroscope mounted on the torso, measuring pitch and roll angular velocities. They applied a threshold algorithm to the peaks in the angular velocity signal, angular acceleration and torso angle change. The second generation of methods is able to classify activities with machine-learning methods, such as decision trees, SVM, kNN, and naïve Bayes. Huỳnh et al. (2007) presented an approach for recognizing daily activities. The movement was sensed by three body-worn accelerometers, while the recognition of 15 low-level and three high-level activities was performed using four approaches: k-means clustering, SVM, nearest neighbor classifier, and hidden Markov models. In the experimental setting the system achieved an accuracy of $69-80\%$ for low-level (for example, sit, eat, walk) and $83-92\%$ for high-level (preparing for work, shopping, housework) activities. Tapia et al. (2007) presented a real-time algorithm for automatic recognition of not only physical activities, but also, in some cases, their intensities, using five wireless accelerometers and a wireless heart rate monitor. The accelerometers were placed at shoulder, wrist, hip, upper part of the thigh and ankle. The features, for example, FFT peaks, variance, energy, correlation coefficients, were extracted from time and frequency domains using a predefined window size on the signal. The classification of activity was done with C4.5 and naïve Bayes classifiers into three groups: postures (for example, standing, sitting), activities (for example, walking, cycling) and other activities (for example, running, using stairs). For these three classes they obtained the recognition accuracy of 94.6% using subject-dependent training and 56.3% using subject-independent training. Kwapisz et al. (2011) used an accelerometer placed on the thigh and compared the results of three classification methods on dynamic activities such as walking, running and jogging. Banos et al. (2013) proposed a hierarchical-weighted classification that combines the majority voting and weighted hierarchical aggregation: at the first level each sensor makes decision about the recognized activity using binary classifiers, while at the next level, weighted majority vote scheme aggregates the decision in order to make the final decision.

Qian et al. (2004) introduced a gesture-driven interactive dance system capable of real-time feedback. They used 41 markers on the body recorded by eight cameras with the frame rate of 120 Hz to construct a human body model. The model was used to extract features such as torso orientation, angles between adjacent body parts etc., which was used to represent different gestures. Each gesture was statistically modeled with a Gaussian random vector defined as the statistical distribution of the features for that gesture. To recognize a new pose, the likelihood of its feature vector given the vector of each known gesture was computed. The new pose was classified as the gesture for which this likelihood was the largest. Experimental results with two dancers performing 21 different gestures achieved gesture recognition rate of 99.3%. Sukthankar and Sycara (2005) presented a system that reconstructs the users posture and recognizes pre-defined behaviors. The data were captured with 43 body markers and 12 cameras with the sampling rate of 120 Hz. They constructed a human body model from the raw marker coordinates, and computed features, for example the angles between body parts, limb lengths, range of motion etc. from the model. Learning was performed using SVM. The method achieved 76.9% accuracy in detecting walking, running, sneaking, being wounded, probing, crouching, and rising. Behavior was defined as a sequence of elementary activities and was modeled with hidden Markov models. The authors defined a number of behavior models and classified a new sequence of activities into the model that fit it best.

Our work follows the second-generation acceleration-based activity recognition, but it demonstrates an approach based on wearable location sensors, where considerable amount of noise is present. In contrast to related work, it performs activity recognition in pipeline; that is, noise removal, activity recognition, removal of spurious activity transitions, and recognition of complex activities. Compared to work by Sukthankar and Sycara (2005) and Qian et al. (2004), our work deals with two orders of magnitude less accurate location system and only four location tags.

Activity recognition based on wearable sensors suffers from some limitations (Chen et al., 2012); that is, most of the sensors need to run constantly and be operated hands-free. Practical issues involve the user acceptability and ability to wear the sensors, while technical issues include size, battery life and ease of use. Moreover, wearable sensors may not be suitable for monitoring activities that include interactions with the environment. As a result, it is often advantageous to combine wearable sensors with embedded sensors, which are described bellow.

Embedded Sensors

Embedded sensors, sometimes referred to as dense sensors, are attached to objects and activities are monitored by detecting object-agent interactions. Using dense sensing, a large number of usually low-cost, miniaturized sensors are deployed in a range of objects and locations within in an environment. This approach is based on the assumption that activities are characterized by the objects that are manipulated during their performance; that is, activities can be recognized from sensor data that monitor agent interactions with the objects in the environment (Chen et al., 2012).

Activity recognition based on embedded sensors has been widely adopted in AAL via smart home paradigm to monitor an inhabitant's movements and environmental events, providing just-in-time context-aware ADL assistance. For example, Storf et al. (2009) studied recognition of ADLs from sensors embedded in the environment. They introduced a multi-agent approach that uses an event-driven activity recognition language to compose atomic activities into high-level activities. The authors report accuracy of higher than 80%. In a similar setting Cook and Holder (2011) applied hidden Markov models for recognition of ADLs and varied the number of sensors used for recognition. The achieved accuracy ranged between 80% and 90%, and dropped below 75% when significant number of sensors was removed.

Different types of sensors and modalities have been in different combinations for activity recognition, and it is impossible to claim that one sensor combination is superior. The suitability and performance are tightly related to the type of activities being assessed and the characteristics of the concrete applications.

2.2 Anomalous and Suspicious Behavior Detection

There are two approaches to detecting deviant behavior (Avrahami-Zilberbrand, 2009): *suspicious* and *anomalous* behavior detection. The first approach assumes a behavior library that encodes *negative behavior*, and thus recognizing observed behavior corresponds to identifying a match in the library. The second approach uses the behavior library in an inverse fashion, meaning that the library encodes only *positive behavior*. When an observed behavior cannot be matched against the library it is considered as anomalous. Several approaches have been proposed to tackle the problem either way. We broadly classify anomalous and suspicious behavior detection in three categories: pattern analysis, transaction analysis, and plan recognition.

2.2.1 Pattern Analysis

Anomalous and suspicious behavior detection from patterns is usually based on visual modalities, such as camera. Trajectories of moving objects have been used to infer anomalous agent paths (Zhang et al., 2004; Vaswani et al., 2005), although image-plan trajectory itself is sensitive to translations, rotations and scale changes. Zhang et al. (2007) proposed a system for a visual human motion analysis from a video sequence, which recognizes unusual behavior based on walking trajectories, namely treading tracks. Two types of line shapes were studied: the closed curve and the spiral line. If preson's treading track takes on one of these shapes, this person is wandering around and is, therefore, suspicious. Lin et al. (2009) described a video surveillance system based on color features, distance features, and a count feature, where evolutionary techniques are used to measure observation similarity. The system tracks each person and classifies their behavior by analyzing their trajectory patterns. This is performed with a hybrid genetic algorithm that uses a Gaussian synapse. Another approach includes behavior patterns based on visual features, for example, Arsić et al. (2007) introduced an approach to visual surveillance of public transportation systems. The system extracts a set of visual low-level features in different parts of the image, and performs a classification with SVMs to detect aggressive, cheerful, intoxicated, nervous, neutral, and tired behavior.

2.2.2 Transaction Analysis

Transaction analysis assumes discrete states/transations in contrast to pattern analysis, which is based on continuous observations. A major research area is intrusion detection (ID) that aims detecting attacks against information systems in general. There are two types of ID systems: signature based and anomaly based. Helman and Liepins (1993) proposed an intrusion detection system that provides a rating for computer activities, demonstrating frequentist estimator and matching rules. Esponda et al. (2004) analyzed trade-offs between positive and negative activity patterns in the library and presented an approach based on partially matching rules. These approaches similarly address the problem of how to decide whether a user's activity is suspicious, but differ significantly in the approach to matching and assessing the behavior. A comprehensive review of ID approaches was recently published by Gyanchandani et al. (2012). Quah and Sriganesh (2008) presented an approach to online-banking fraud detection based on persons' spending behaviors. Their approach makes use of a self-organization map to learn persons' spending patterns, while neural networks filter any unusual events and analyze the person behavior in order to detect fraud. In addition, Alexandre (1997) proposed a system based on the keyboard signature behavior recognition, which is more difficult to copy or fake than a fingerprint or a smart card. The presented technique implements a neural network, which is evaluated in terms of efficiency and performance.

Our work leverages ideas by Helman and Liepins (1993) and Esponda et al. (2004) to establish a formal detection framework based on behavior patterns and analyze detection errors. On this basis, we extend the framework to formally address repeated behavior detection and specify conditions any reasonable detector should satisfy.

Furthermore, AAL applications based on wearable sensors also fit to transaction analysis, since sensing is typically event based. Lymberopoulos et al. (2008) proposed a system for automatic extraction of the users' spatio-temporal patterns from the sensor network deployed inside their home. The proposed method, based on location, time and duration, was able to extract frequent patterns using the Apriori algorithm and to encode the most frequent patterns in the form of a Markov chain, while our work uses the location and the activity performed by the user to build a model of normal behavior and detect anomalous behavior patterns. Another area of related work includes hidden Markov models (HMMs) (Rabiner, 1989) that are widely used in traditional activity recognition for modeling a sequence of actions. Brand et al. (1997) introduced coupled HMMs as an extension with multiple hidden interacting chains that are able to model interactive behavior. Duong et al. (2005) focused on the duration of activities and introduced switching hidden semi-Markov models that provide probabilistic constraints over the duration of plans, and applied them to the detection of anomalies in the activities of daily living. Monekosso and Remagnino (2010) used embedded sensors and also addressed the problem of anomalous behavior detection. The output of the sensors was directly used to train a HMM model based on normal observations. If the likelihood that a new observation was generated by the trained model was low, the behavior was considered abnormal. Our work first recognizes the user's activities from sensor data and then combines them with spatial information. Compared to HMMs, it does not require an estimation of the parameters in the learning phase. Although widely used, HMMs may become inadequate when actions are more complex or have long-term temporal dependencies (Koller and Friedman, 2009).

Lee et al. (2004) proposed a fuzzy-association analysis of an individual's daily patterns based on an infrared location sensor and activity sensor groups (for example, sleeping, eating, leisure sensor group). They defined two fuzzy membership functions: start time (for example, dawn, morning) and duration (for example, short, medium), and transformed a sequence of activities using these two functions to categorical attributes. Afterwards, the Apriori algorithm was applied to the dataset, searching for activity patterns. The authors suggest that the behavioral pattern changes indicate that the person is not well. Lymberopoulos et al. (2008) proposed a system for automatically extracting person spatio-temporal patterns from a home-deployed sensor network. The proposed method, based on location, time, and duration, was able to extract patterns using the Apriori algorithm and to encode the most frequent ones in a Markov chain.

In contrast to related work, we propose a presentation that encodes activity dynamics; that is, it explores the relations between spatial information and activities to capature behavior dynamics in a specific time period. Our work first recognizes the user's activities from sensor data and then combines them with spatial information.

2.2.3 Plan Recognition

Plan recognition focuses on a mechanism for recognizing the unobservable state of an agent, given observations of its interaction with its environment (Avrahami-Zilberbrand, 2009). Most existing investigations assume discrete observations in a form of activities. To perform anomalous and suspicious behavior detection, plan recognition algorithms may use a hybrid approach: a symbolic plan recognizer is used to filter consistent hypotheses, passing them to an evaluation engine, which focuses on ranking.

Avrahami-Zilberbrand and Kaminka (2007) presented Utility-based Plan Recognition (UPR) that introduces utility to the observer in selecting the recognition hypotheses. The main strength of UPR is that it can incorporate an observer's bias to events with a low likelihood, for example, the a-priori probability for planting a bomb is very low, but detecting it has a high expected utility. We further discuss this approach in Section 6.3.3. Geib and Goldman (2009) presented PHATT, a probabilistic approach based on tree grammars able to cope with interleaved goals, partially ordered plans, and failed observed actions. Sukthankar and Sycara (2008) addressed plan recognition for multiagent teams, where plans were ordered by linear accumulation of observed actions consistent with the plan.

Our work leverages UPR approach to perform repeated behavior detection. The notion of utility, which is assigned to each plan step by the observer, is extended with the notion of utility function that generalizes utility-based plan recognition with arbitrary utility functions. This allows to assign utility to repeated plan steps according to agent past behavior.

Part I

Deviant Behavior Detection

3 Activity Recognition

Activity recognition is an underpinning task in behavior analysis. It transforms sensor data to a higher-level description of behavior primitives. This chapter presents a pipeline for recognizing an agent's atomic activities from multidimensional, sequential, spatio-temporal data. We first formulate the problem and discuss the basic ideas, followed by a description of the pipeline components, including noise filtering (Section 3.2), designing feature vectors and model learning (Section 3.3), and providing recognized activity continuity (Section 3.4). In addition, we present approaches for recognizing compound activities and activities that result from interactions among agents (Section 3.6).

3.1 Problem Statement and Basic Ideas

Observing an agent's atomic activities is relatively straightforward in domains with discrete states. For example, observing a person on a computer terminal consists of typing a command that changes the state; hence, an atomic action corresponds to a command. Other continuous domains, where an agent is observed with sensors providing sequential numerical data, require more steps, since these sequences are not automatically labeled with activities. The main problem is how to segment sequences into atomic activities.

Consider an environment where the movement of an agent is observed with several sensors providing measurements at each time step t.

Definition 1. Observation vector $\mathbf{x}_t$ *is a multi-dimensional signal vector containing stochastic values from each sensor at a given time point* t.

At this point, we assume that it is possible to construct an observation from all the sensors regardless of the frequency with which a particular sensor provides measurements.

Definition 2. Observation sequence $\mathbf{X}$ *consists of* T *observation measurements such that* $\mathbf{X} = \{\mathbf{x}_t | 1 \leq t \leq T\}$.

Given a finite set of possible activities $\mathbb{A} = \{a_1, ..., a_K\}$, our goal is to automatically segment an observation sequence $\mathbf{X} = \{\mathbf{x}_1, ..., \mathbf{x}_T\}$ into a sequence of activities $\{a_1, ..., a_T\}$. In the literature, activity is often referred to as action, activity, complex activity, compound task, goal, or plan. The main difference is how many observations are used to assign an activity. We define activity as behavior in a specific time span, as follows:

Definition 3. Activity $a_{i,j} \in \mathbb{A}$ *describes an action as behavior caused by an agent in a particular situation limited by time span* $i \leq t \leq j$ *that explains observations* $\mathbf{x}_i, ..., \mathbf{x}_j$*, where* $\mathbb{A}$ *is a set of possible activities.*

A special case is when activity corresponds to a single observation; that is, $a_{i,i}$. We denote such activity as atomic action.

Definition 4. Atomic action $a_t \in \mathbb{A}$ *at time step* t *is an activity from a set of possible activities* $\mathbb{A}$ *assigned to an observation* $\mathbf{x}_t$ *with function*

$$f : \mathbb{R}^{|\mathbf{x}|} \to \mathbb{A}.$$

However, in general, the number of observations described by an activity is not fixed, since different behaviors require a different number of observations, and behavior itself can be presented at different granularities. For example, behavior *breakfast routine* can be an activity itself, or it can be segmented into several activities, for example, *making coffee*, *putting food on the table*, *eating*, and *cleaning*.

Labeling multi-dimensional time-series sensor data is inherently more complex than classifying traditional, nominal data that contain little noise. First, each observation is temporally connected to the previous and next observations, making it very difficult to apply a straightforward classification of a single observation only. Second, the data obtained by sensors at different time points are stochastic due to sensor noise, environmental disturbances, and many other reasons. Moreover, an activity can comprise various sub-activities executed in different manners, resulting in high intra-class differences. Finally, all these reasons make an activity recognition model imprecise, resulting in unseen observation vectors being mislabeled. Therefore, it is highly desirable to ensure continuity and consistency in the recognized activity sequence.

To deal with the above-mentioned challenges, we propose an activity-recognition pipeline, referred to as ARPipe, as shown in Figure 3.1. We first devise a noise removal phase, which is strongly tightened with the type of sensors deployed in the domain; then we show an approach based on location-based sensors attached to the human body. The next phase extracts domain-dependent features from a set of observations and constructs a feature vector, which is used in the next step. Based on the feature vector, an activity recognition model assigns an atomic action to each observation. Finally, transitions between activities that cannot occur in reality are removed in the last step.

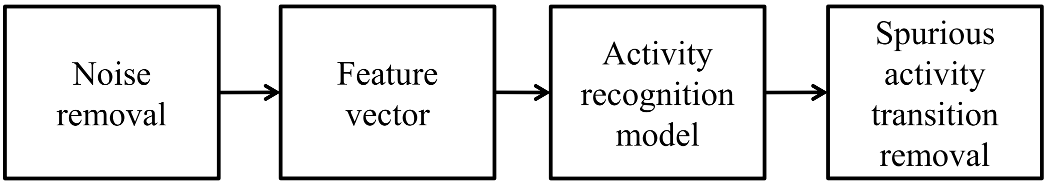

Noise removal
Feature vector
Activity recognition model
Spurious activity transition removal

Figure 3.1: ARPipe, an activity recognition pipeline.

The activity recognition model is constructed with a supervised learning approach, which consists of training and classification steps. In the training step, a set of labeled data is provided to train the model. The second step is used to assign a label to new, unseen data by the trained model. The data in both phases must be pre-processed with the same set of tools, such as filtering and feature vector computation.

The post-processing phase; that is, spurious activity removal, can also be a model itself and, hence, also requires a learning step. In this case, the pre-processing step also includes activity recognition. Therefore, it is important that the dataset used for training is not the same as that used for the training activity recognition model.

The first two phases in the proposed pipeline are domain-dependent, while the last two phases are general. We demonstrate our methods in the ambient-assisted living domain (see Chapter 8) to recognize activities performed by an elderly person in a home environment, wearing location-based tags that provide absolute three-dimensional coordinates. The goal is to label an observation vector with eight atomic actions. Specifically, we present two approaches for filtering and feature vector construction to illustrate how they can be used to robustly recognize activities in the presence of noise, clutter, and human action execution variability.

3.2 Dealing with Noise

An important sensor technology that makes human activity recognition possible relies on real-time location systems (RTLSs). They provide three-dimensional coordinates of tags attached to the human body. High-fidelity optical system, such as Vicon (2009) and SMART (eMotion, 2009), provide accurate measurements (±2 mm), but often include outliers due to marker occlusion and mislabeling. Furthermore, they require a line-of-sight between the body-attached tags and the cameras. They are good for lab use, but fail in real-world applications as they are usually too expensive, hard to install, and have limitations, such as line-of-sight or a confined operational area. More affordable systems rely on radio technology, which is less obtrusive and cheaper, but less accurate. Systems based on ultra-wideband technology (UWB), such as Ubisense (Steggles and Gschwind, 2009), achieve ±15 cm accuracy in an ideal setting, which makes human activity recognition challenging. The main problem addressed in this section is how to de-noise human-motion trajectories captured with UWB RTLS in order to improve activity recognition.

We demonstrate noise filtering on Ubisense, a commercially available localization system. Ubisense allows local positioning by tracking a set of reasonably small tags, which are attached to a person's body. A sampling frequency of around 10 Hz can be achieved with up to four tags attached to a person simultaneously. Each tag maintains radio contact with a stationary sensor (for example, mounted on the wall). These sensors and tags communicate using UWB radio technology. Both the time arrival difference and the radio signal arrival angle are used to calculate the tag position. In a typical open environment, a location accuracy ±15 cm can be achieved across 95% of the readings. However, in real-life scenarios the accuracy occasionally exceeds ± 200 cm, which represents quite a challenge for preprocessing and filtering.

De-noising human motion captured with UWB sensors raises several challenges. First, motion-capture data may contain a certain percentage of missing values due to packet loss, temporal sensor disability, low battery, etc. Second, due to sensor noise and environment disturbances, motion-capture data often include outliers and unstable measurements, which corrupt body posture reconstruction. This results in a violation of physical body constraints as well as spatio-temporal body constraints, which in turn introduces additional noise into the activity recognition models. Finally, essential activity recognition features that are computed from noisy measurements (for example, velocity, acceleration) may have an integral error term; that is, the error accumulates over time.

In this section, we propose an efficient approach for de-noising human-motion trajectories that not only filters corrupted motion data, but also enforces the human body's spatio-temporal constraints and enables more accurate feature computation. The key idea is to construct a series of filters that addresses the above-mentioned challenges.

3.2.1 Dealing with Outliers

Median filter is a non-linear filter that can suppress impulsive, isolated noise without blurring sharp changes in the signal (Yin et al., 1996). The filter consists of a sequential sample window with odd length $w = 2n + 1$. At each time step t, the filter returns the median of the elements in the window:

$$x'_t = \text{median}(x_{t-n}, ..., x_t, ..., x_{t+n}). \tag{3.1}$$

The only parameter of the median filter is the window length w, which introduces a delay of length $\lfloor w/2 \rfloor$. A larger window size may smooth the signal too much, while a smaller window size may not remove the high density noise. A common approach is to choose a window length such that desired signal features are preserved while attenuating noise.

In our case, we apply the median filter at each tag, separately for each dimension. The filter removes isolated spikes in the signal, while parts with high oscillation remain unsuppressed.

3.2.2 Missing Values and Velocity Estimation

Motion capture data contains missing values due to packet loss or delay during transmission, sensor failure, corrupted packets, etc. Our first goal is, hence, to fill the missing values. Furthermore, we would like to estimate additional quantities such as velocity. For this task, we use the recursive linear Kalman filter (Kalman, 1960) for optimally estimating he system's state, assuming that the underlying system is a linear dynamical system and that all measurement errors have a multivariate Gaussian distribution. The underlying system, that is, human body, is assumed to be reasonably approximated by a linear dynamical system. Even though the original measurement error distribution is not Gaussian, the median filtering removes signal spikes, which results in a measurement error distribution that better resembles Gaussian distribution.

The Kalman filter performs the following three tasks: smoothing of the UWB measurements, estimating the velocities of tags, and predicting the missing measurements. We defined filter state as a six-dimensional observation vector $\mathbf{x}_t$ that includes positions and velocities in each of the three dimensions at time t, $\mathbf{x}_t = [p_{x,t}, p_{y,t}, p_{z,t}, v_{x,t}, v_{y,t}, v_{z,t}]^\intercal$.

The next state is estimated from the previous state as follows:

$$\mathbf{x}_{t+1} = \mathbf{F}\mathbf{x}_t + \mathbf{B}\mathbf{u}_t + \mathbf{w}_t, \tag{3.2}$$

where $\mathbf{F}$ encodes the linear dynamical system, $\mathbf{B}$ is a control matrix and $\mathbf{w}_t$ is noise covariance matrix. In our case, the Kalman update is simplified to Equation (3.3). The next state is calculated as a sum of the previous position and a product of the previous velocity and the time between the consecutive measurements Δt for each direction separately. The velocities remain constant. The measurement noise covariance matrix was set based on UWB system specification, while the process noise covariance matrix was fine-tuned experimentally.

$$\begin{bmatrix} p_{x,t+1} \\ p_{y,t+1} \\ p_{z,t+1} \\ v_{x,t+1} \\ v_{y,t+1} \\ v_{z,t+1} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & \Delta_t & 0 & 0 \\ 0 & 1 & 0 & 0 & \Delta_t & 0 \\ 0 & 0 & 1 & 0 & 0 & \Delta_t \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} p_{x,t} \\ p_{y,t} \\ p_{z,t} \\ v_{x,t} \\ v_{y,t} \\ v_{z,t} \end{bmatrix} + \mathbf{w}_t. \tag{3.3}$$

3.2.3 Spatio-Temporal Body Constraints

Up to this point, each tag was considered as a separate measurement. In reality, the tags are attached to a human body, which implies a set of tag position constraints. In activity recognition, it is expected that a set of measurements resembles human body proportions as well as spatio-temporal patterns in natural human motion. We construct a filter based on iterative constraint relaxation that: (i) projects measured values in a valid domain; (ii) applies human body constraints to the measured positions; and (iii) constrains spatio-temporal motion patterns.

In the first step, we make an assumption about valid measurement domain. For example, we expect all the measurements to be within a room, that is, cuboid, bounded with two extreme points $\mathbf{p}_X$ and $\mathbf{p}_Y$ (assuming the coordinate system is aligned with the room). To keep the measurement $\mathbf{p}_t$ within the expected bounds, it has to be translated to an edge (in case it is not already within the cuboid) as shown in Figure 3.2. The update step is:

$$\mathbf{p}'_t = \min(\max(\mathbf{p}_t, \mathbf{p}_X), \mathbf{p}_Y). \tag{3.4}$$

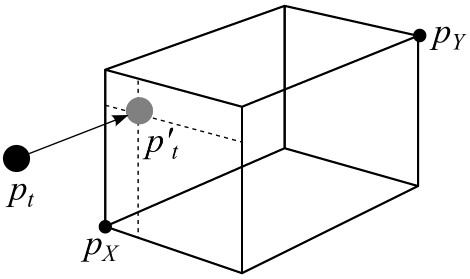


Figure 3.2: All the measurements are bounded within a cuboid.

We model the human body using rigid-body components, which assume that there is no deformation. Rigid-body components are connected to each other with joints and form an *articulated rigid body* that approximates the human body as shown in Figure 3.3. The distance between any two connected joints is constant regardless of external forces. Note that at this point we do not pose any joint constraints.

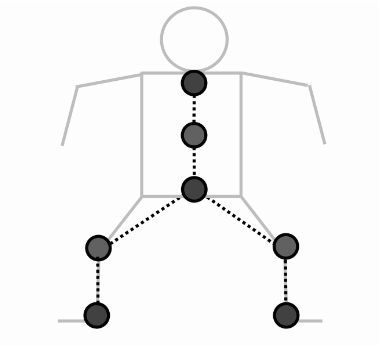


Figure 3.3: Human body is modeled with an articulated rigid body.

In our case, the four RTLS tags provide the joint positions (ankles, waist, and chest), but do not allow reconstructing the skeleton displayed in Figure 3.3 since the knees and abdomen are missing.

The missing joints are reconstructed as follows: suppose we have two points A and C with known positions and a joint B that interconnects A and C, with an unknown position. Since the distances $r_A = d(A, B)$ (between A and B) and $r_c = d(C, B)$ are known, the point B then lies at the intersection of two spheres, centered at A with radius r_A and at C with radius r_C.

In general, there are three cases when the measurements are obtained: (i) $r_A + r_B = d$, that is, the intersection includes a single point; (ii) $r_A + r_B > d$, that is, there is no solution; and (iii) $r_A + r_B < d$, that is, the intersection consists of a circle. In the second case we position the point B on a line between points A and C so that the distances between points is in the same proportion as the lengths of r_A and r_B. In the third case, we proceed as follows: to calculate the position of the point B, we use a new coordinate system in which the first sphere is centered at the origin and the second sphere is centered at a point on the positive x-axis, at distance d from the origin, as shown in Figure 3.4. Subtracting the sphere equations, we find a set of points representing a circular intersection of the spheres:

$$x = \frac{d^2 - {r_C}^2 + {r_A}^2}{2d}, \tag{3.5}$$

$$y^2 + z^2 = {r_A}^2 - (\frac{d^2 - {r_C}^2 + {r_A}^2}{2d})^2. \tag{3.6}$$

We are not interested in the exact position of B, hence we pick an arbitrary point from the circle and transform it to the original coordinate system. As explained below, the distance between joints is enforced with Equations (3.8) and (3.9).

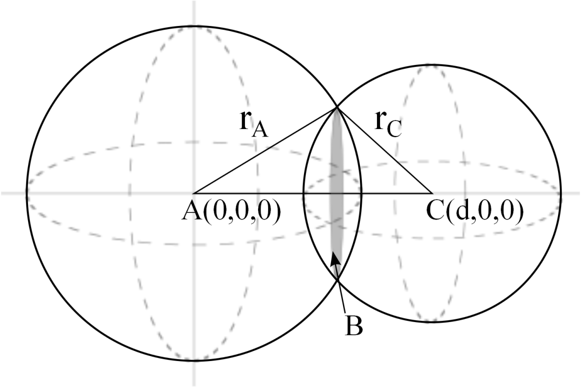


Figure 3.4: The result of sphere-sphere intersection is a circle.

Once we have all the joint positions we can introduce constraints between the connected pairs. For example, suppose the true distance between joints A and B is r_A, that is,

$$\|\mathbf{p}_A - \mathbf{p}_B\| = r_A. \tag{3.7}$$

If measurements p_A and p_B violate the constraint given by Equation (3.7), the position of both points is adjusted (Jakobsen, 2001). Each point is translated along the line connecting the points for half of the error defined as the difference between the measured and the true distance as shown in Figure 3.5. The update is:

$$\mathbf{p}'_A = \mathbf{p}_A + \frac{\|\mathbf{p}_B - \mathbf{p}_A\| - r_A}{2\|\mathbf{p}_B - \mathbf{p}_A\|}(\mathbf{p}_B - \mathbf{p}_A), \tag{3.8}$$

$$\mathbf{p}'_B = \mathbf{p}_B - \frac{\|\mathbf{p}_B - \mathbf{p}_A\| - r_A}{2\|\mathbf{p}_B - \mathbf{p}_A\|}(\mathbf{p}_B - \mathbf{p}_A). \tag{3.9}$$

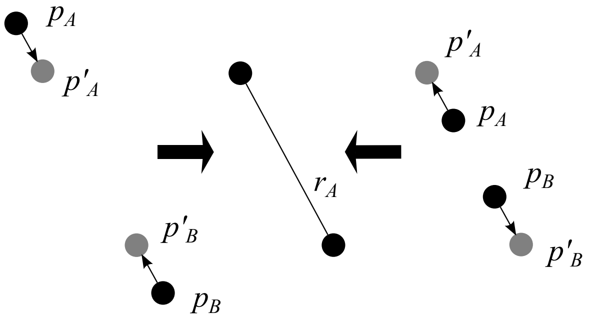


Figure 3.5: Move the points $\mathbf{p}_A$ and $\mathbf{p}_B$ to match the constraint given by Equation (3.7).

In addition to the constraints introduced by human body proportions, we also consider physical motion constraints such as velocity and acceleration, of limbs. Suppose that a m/s^2

is the greatest possible acceleration of an ankle. This implies that it can travel at most $l = (v_{t-1} + a\Delta t/2)\Delta t$ meters in time interval Δt, where $1/\Delta t$ is the sampling frequency. Hence, the next ankle's position $\mathbf{p}_t$ is limited with a sphere with radius l, that is,

$$\|\mathbf{p}_t - \mathbf{p}_{t-1}\| \leq l. \tag{3.10}$$

In case the new position is outside the sphere, the position is translated on the edge of the sphere in the direction of the measurement. The update step is:

$$\mathbf{p}'_t = \mathbf{p}_t + \frac{l(\mathbf{p}_t - \mathbf{p}_{t-1})}{\|\mathbf{p}_t - \mathbf{p}_{t-1}\|}. \tag{3.11}$$

Finally, all the constraints are put together. Consider $\mathbf{C} = \{\Phi_i\}$ as a set of constraints, where $\Phi(\mathbf{p})$ applies the update step on point $\mathbf{p}$ using Equation (3.4); that is, $\mathbf{p}' \leftarrow \Phi(\mathbf{p})$, while $\Phi(\mathbf{p}_A, \mathbf{p}_B)$ applies the update step on both points A and B using Equations (3.8) and (3.9); that is, $\mathbf{p}'_A, \mathbf{p}'_B \leftarrow \Phi(\mathbf{p}_A, \mathbf{p}_B)$. If a constrained between points A and B is not present, then $\Phi(\mathbf{p}_A, \mathbf{p}_B)$ does not alter the corresponding points. The algorithm below takes the set of constraints and its value updates as an input and iteratively updates the values until the convergence threshold τ_c or maximal number of iterations k is reached.

Algorithm 3.1 Iterative constraint relaxation.

```
Require: set of constraints $\mathbf{C}$, set of points $\mathbf{P}$, maximal number of iterations $k$, convergence threshold $\tau_c$
Ensure: set of points $P$
  repeat
    $\Delta = 0$
    for $\mathbf{p} \in \mathbf{P}$ do
      for $\mathbf{q} \in \mathbf{P}$ do
        $\mathbf{p}', \mathbf{q}' \leftarrow \Phi(\mathbf{p}, \mathbf{q})$
        $\Delta \leftarrow \Delta + |\mathbf{q} - \mathbf{q}'|$
        $\mathbf{q} \leftarrow \mathbf{q}'$
      end for
      $\mathbf{p}' \leftarrow \Phi(\mathbf{p})$
      $\Delta \leftarrow \Delta + |\mathbf{p} - \mathbf{p}'|$
      $\mathbf{p} \leftarrow \mathbf{p}'$
    end for
    $k \leftarrow k - 1$
  until $k > 0$ and $\Delta > \tau_c$
```

An example of filter effects is shown in Figure 3.6, which shows x (top), y (middle) and z (bottom) coordinates for a tag attached to the waist for $T = 600$ time steps (one time step lasts approximately 1/8 s). The vertical axis corresponds to meters. The blue line represents the original location measurements, the green line represents the median filter result, and the red line represents the Kalman filtering and spatial body constraints results.

3.3 Feature Vector

Finding an appropriate representation of the person's activities is probably the most challenging part of activity recognition. The behavior needs to be represented with simple and general features, so that the model using these features will also be general and work well

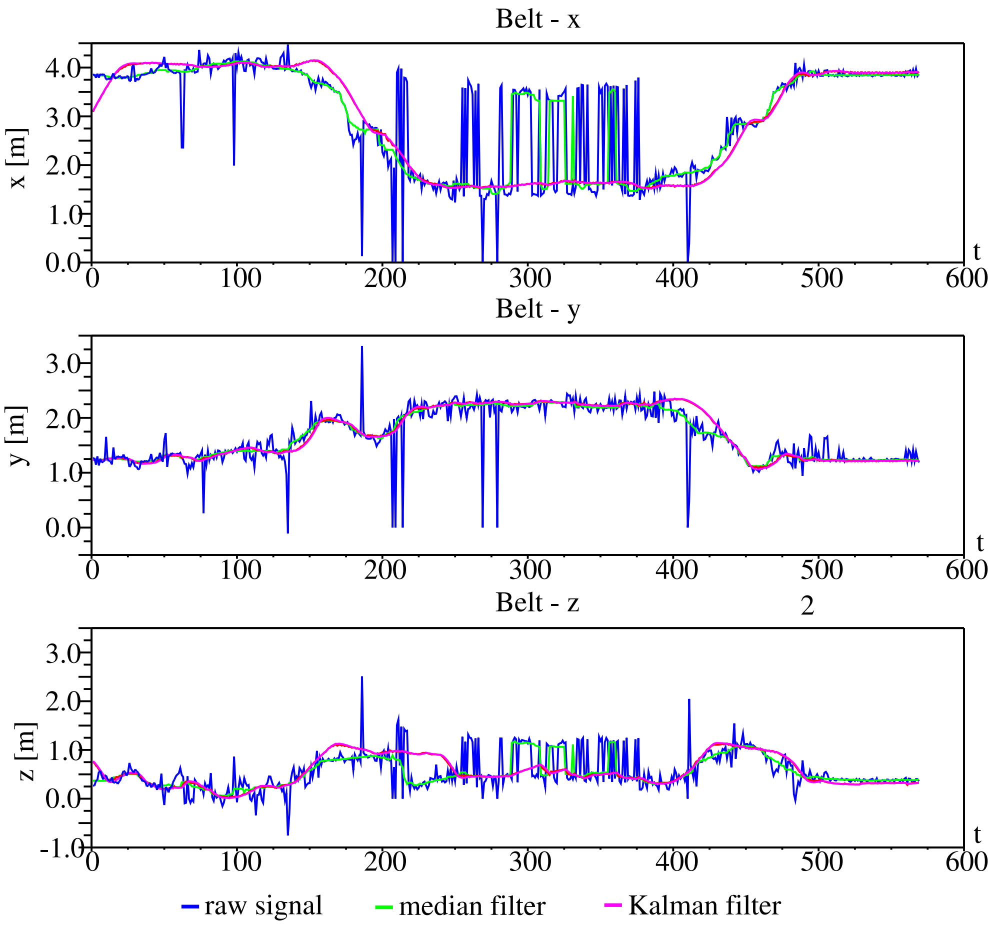


Figure 3.6: Filtered coordinates x (top), y (middle) and z (bottom) of a tag attached to the waist.

on behaviors different from those in the learning set. In fact, it is not difficult to design features specific to captured observations in a training set; such features would work well on them. However, since the training set captures only a part of the whole range of human behavior, overly specific features would likely fail on general behavior.

Definition 5. Feature vector $\mathbf{f}_t$ *at time step* t *is a vector of descriptors obtained from observation* $\mathbf{x}_t$*.*

Feature $\mathbf{f}_t$ can hence contain values from observation as well as additional values computed from observation $\mathbf{f}_t$ or other observations. In general, the feature vector can be interpreted as observation extended with additional descriptors that capture targeted behavior.

3.3.1 Features

We propose three sets of features describing the person behavior in a selected domain. We demonstrate a possible feature set on a kinematic model of a human with 12 points as shown in Figure 3.7 to illustrate potential variety. In practice, however, it is possible to use a reduced feature set. First, reference features are expressed in the reference coordinate system, which is fixed with respect to the person's environment. Second, body features are expressed in a coordinate system affixed to the person's body. Third, angle features are the angles between adjacent body parts.

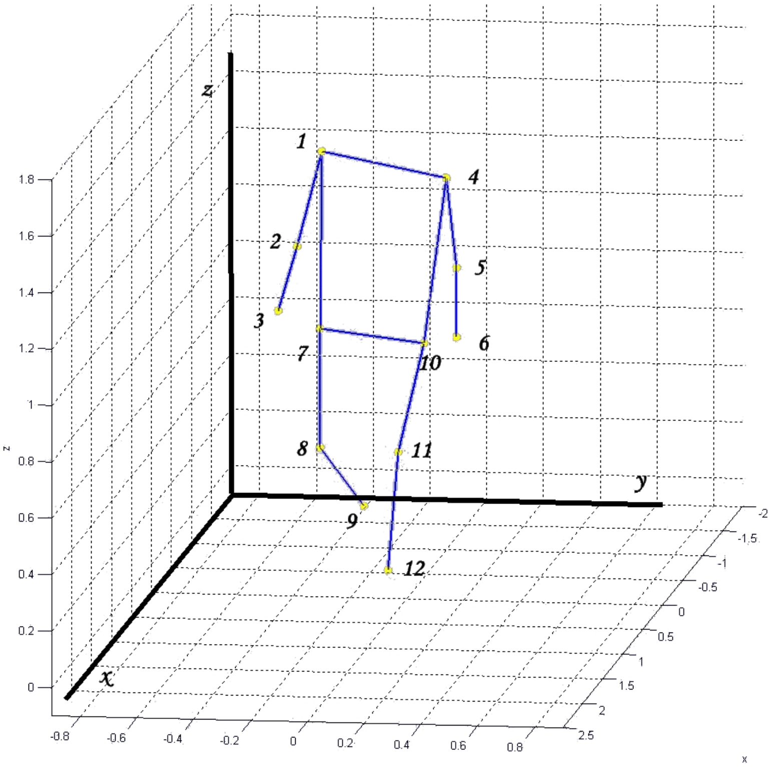


Figure 3.7: Kinematic model of a human body.

Reference Features

When selecting reference features, the x and y coordinates are usually ignored, since these coordinates describe the person's location in the environment, but the activities of interest can generally take place at any location. A reasonable set of features can include $z_t^{(i)}$ coordinate of sensor tag i at time t, the velocity of sensor tag in z direction $v_t^{(i)}$, the absolute distance $d_t^{(i,j)}$between sensor tags i and j, and others.

Body Features

Body features are expressed in a coordinate system affixed to the person's body. This makes it possible to observe x and y coordinates of the person's body parts, since these coordinates no longer depend on locations within the environment.

The body coordinate system is shown in Figure 3.8. Its origin $\mathbf{o}$ is at the mid-point of the line connecting the hip tags ($\mathbf{p}_7$ and $\mathbf{p}_{10}$ in Figure 3.8). This line also defines the y axis, which points towards the left hip. The z axis is perpendicular to the y axis, touches the line connecting both shoulder tags ($\mathbf{p}_1$ and $\mathbf{p}_4$ in Figure 3.8) at point $\mathbf{p}_z$, and points upwards. The x axis is perpendicular to the y and z axes and points forwards.

In order to translate reference coordinates into body coordinates, Equation (3.12) expresses the origin $\mathbf{o}$ and basis $(\mathbf{i}, \mathbf{j}, \mathbf{k})$ of the body coordinate system in the reference coordinate system, which gives us the basis vector $\mathbf{j}$:

$$\mathbf{o} = \frac{\mathbf{p}_7 + \mathbf{p}_{10}}{2}, \tag{3.12}$$

$$\mathbf{j} = \frac{\mathbf{p}_7 - \mathbf{o}}{|\mathbf{p}_{10} - \mathbf{o}|}. \tag{3.13}$$

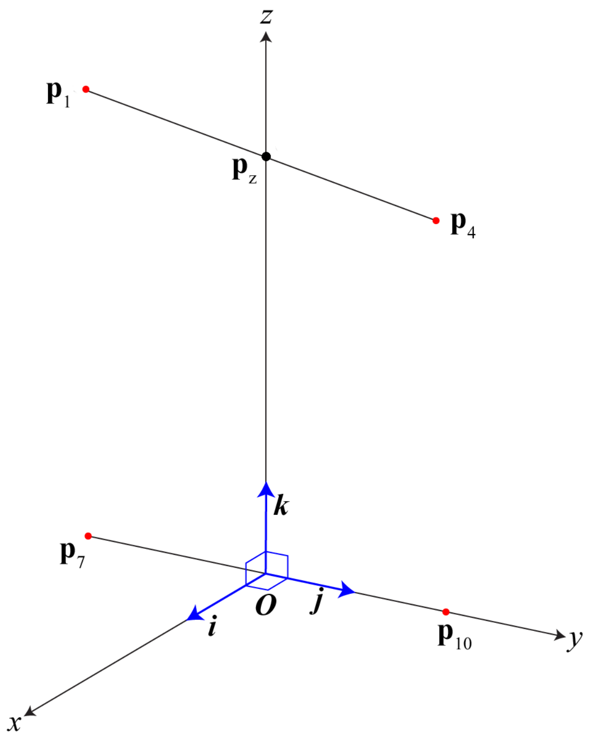


Figure 3.8: The body coordinate system.

To obtain the basis vector $\mathbf{k}$, Equation (3.14) is first used to calculate $\mathbf{p}_z$; Equation (3.16) then gives us $\mathbf{k}$:

$$\mathbf{p}_z = \mathbf{p}_7 + a(\mathbf{p}_4 - \mathbf{p}_7), \tag{3.14}$$

$$a = \frac{(\mathbf{p}_1 - \mathbf{o}) \cdot (\mathbf{p}_{10} - \mathbf{p}_7)}{(\mathbf{p}_4 - \mathbf{p}_1) \cdot (\mathbf{p}_{10} - \mathbf{p}_7)}, \tag{3.15}$$

$$\mathbf{k} = \frac{\mathbf{p}_7 - \mathbf{o}}{|\mathbf{p}_{10} - \mathbf{o}|}. \tag{3.16}$$

Finally, we obtain basis vector $\mathbf{i}$ using Equation (3.17):

$$\mathbf{i} = \mathbf{j} \times \mathbf{k}. \tag{3.17}$$

To translate a point $\mathbf{p}$ coordinates in the reference coordinate system into the body coordinate systems, Equation (3.18) is used. The vector $\mathbf{p}_R = (x_R, y_R, z_R, 1)$ corresponds to the point $\mathbf{p} = (x, y, z)$ in the reference coordinate system. The vector $\mathbf{p}_B = (x_B, y_B, z_B, 1)$ corresponds to the point $\mathbf{p} = (x_B, y_B, z_B)$ in the body coordinate system. Matrix $\mathbf{T}_{R \to B}$ is the transformation matrix from the reference to the body coordinate system.

$$\mathbf{p}_B = \mathbf{T}_{R \to B}\mathbf{p}_B^{\intercal}, \tag{3.18}$$

$$\mathbf{T}_{R \to B} = \begin{bmatrix} \mathbf{i}_x & \mathbf{i}_x & \mathbf{i}_x & -\mathbf{o} \cdot \mathbf{i} \\ \mathbf{j}_x & \mathbf{j}_x & \mathbf{j}_x & -\mathbf{o} \cdot \mathbf{i} \\ \mathbf{k}_x & \mathbf{k}_x & \mathbf{k}_x & -\mathbf{o} \cdot \mathbf{i} \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

Based on the above equations, a reasonable set of body attributes may include the tag's body features, absolute velocity, and the angles of movement with respect to the z axis and xz plane, and others.

Angle Features

Once the body coordinate system is obtained, it is possible to compute an advanced set of features describing relative angles between different body parts represented by quaternions. Unit quaternions provide a convenient mathematical notation for representing object orientations and rotations in three dimensions. Compared to Euler angles, they are simpler and avoid the gimbal lock problem. Compared to the rotation matrices, they are more efficient and numerically stable.

This thesis will not go into further details on angle equations; the reader is referred to Luštrek et al. (2008) for details. A reasonable set of attributes may include the angles of the left and right elbow, the left and right knee, the left and right shoulder (represented by quaternions), the left and right hip (also represented by quaternions), and others.

3.3.2 Canonical Representation

Due to its continuous nature, determining the exact transition points between activities is difficult. We address this issue by combining features into short, overlapping time windows during which we assume that a single activity is dominant. We denote this representation as canonical form that represents the equivalence classes. To test whether two activities in specific time interval are equivalent, it suffices to test their canonical forms for equality. A canonical form thus enables classification and gives a distinguished (canonical) representative of an action.

In practical terms, one wants to be able to recognize the canonical forms; that is, overlapping windows. Each window is classified independently as described in the next section (Section 3.3.3).

More formally, we define:

- w is window length that contains an even number of observations,
- i is an index over $|w|$ elements in window, and
- j is an index over W overlapping windows.

Given a sequence of observations $\{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_T\}$, we first compute the sequence of features $\{\mathbf{f}_1, \mathbf{f}_2, ..., \mathbf{f}_T\}$. Then, for constants m and n, where m is the number of elements before t, and n after t; that is, $m + n = |w| + 1$, we construct W_j, $1 \leq j \leq (|T| - |w| + 1)$ windows such that

$$W_j = \{\mathbf{f}_i | j - m \leq i \leq j + n\}. \tag{3.19}$$

Each W_j is labeled with a dominant activity from $\mathbb{A}$ that is assigned to the feature vector $\mathbf{f}_t$. Note, that Equation (3.19) allows $\mathbf{f}_t$ to adopt any position in W_j; in practice it is placed at the beginning ($m = 0$), middle ($m = n$), or at the end ($n = 0$). An example for $w = 9$ and $m = n = 4$ is shown in Figure 3.9. First, each feature vector is constructed by three sets of attributes. Then, the vector is presented in canonical representation with a window. The end of the window is labeled with the dominant activity.

The intuition behind the canonical representation is that an agent's actions are continuous and, hence, defined not only by the current values but also by the values in a certain time span. So far, the features were expressed in corresponding coordinate systems at each time step t. However, the features $\mathbf{f}_i$ at time steps $i \neq t$ in the canonical representation can be expressed in the coordinate system belonging to the feature at time step t. This explicitly captures the changes between time steps within the interval.

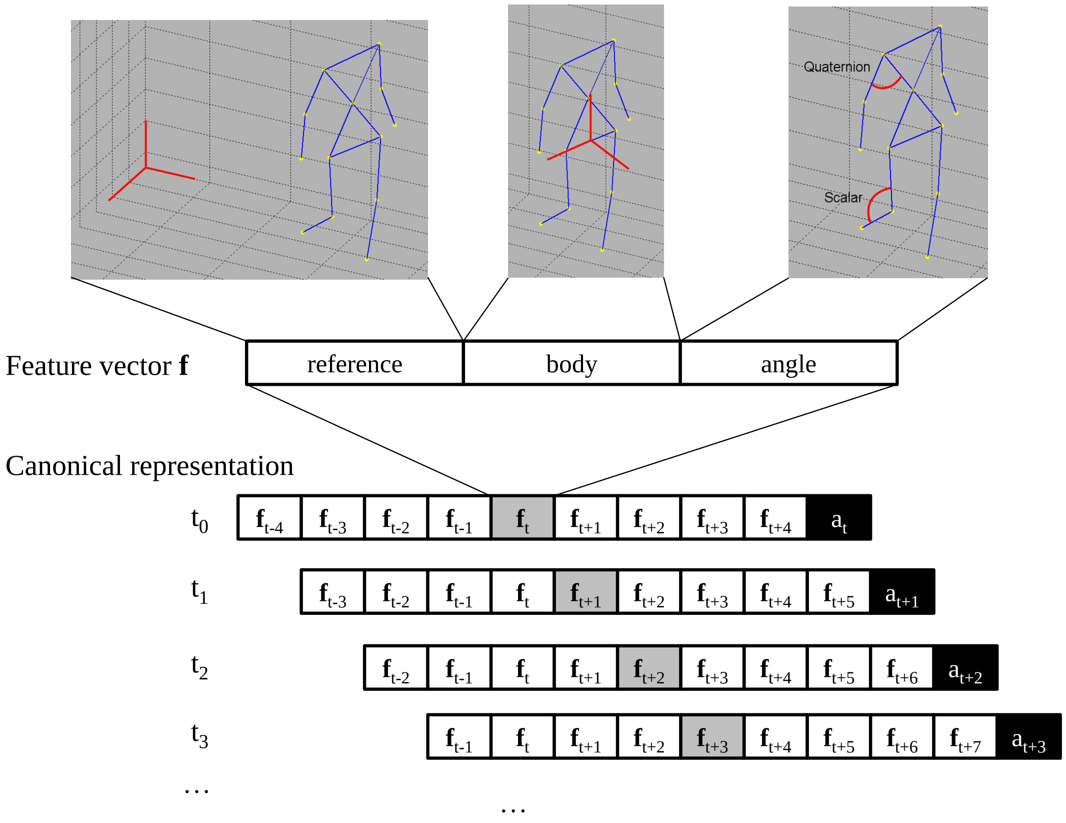

Quaternion
Scalar
Feature vector **f**
reference
body
angle
Canonical representation
t_0 $\mathbf{f}_{t-4}$ $\mathbf{f}_{t-3}$ $\mathbf{f}_{t-2}$ $\mathbf{f}_{t-1}$ $\mathbf{f}_{t}$ $\mathbf{f}_{t+1}$ $\mathbf{f}_{t+2}$ $\mathbf{f}_{t+3}$ $\mathbf{f}_{t+4}$ a_{t}
t_1 $\mathbf{f}_{t-3}$ $\mathbf{f}_{t-2}$ $\mathbf{f}_{t-1}$ $\mathbf{f}_{t}$ $\mathbf{f}_{t+1}$ $\mathbf{f}_{t+2}$ $\mathbf{f}_{t+3}$ $\mathbf{f}_{t+4}$ $\mathbf{f}_{t+5}$ a_{t+1}
t_2 $\mathbf{f}_{t-2}$ $\mathbf{f}_{t-1}$ $\mathbf{f}_{t}$ $\mathbf{f}_{t+1}$ $\mathbf{f}_{t+2}$ $\mathbf{f}_{t+3}$ $\mathbf{f}_{t+4}$ $\mathbf{f}_{t+5}$ $\mathbf{f}_{t+6}$ a_{t+2}
t_3 $\mathbf{f}_{t-1}$ $\mathbf{f}_{t}$ $\mathbf{f}_{t+1}$ $\mathbf{f}_{t+2}$ $\mathbf{f}_{t+3}$ $\mathbf{f}_{t+4}$ $\mathbf{f}_{t+5}$ $\mathbf{f}_{t+6}$ $\mathbf{f}_{t+7}$ a_{t+3}
...
...

Figure 3.9: An example of 10 feature vectors in canonical representation.

3.3.3 Activity Recognition Model

Once feature vectors are represented in the canonical form, it is possible to apply standard techniques for supervised classification, including feature selection, feature discretization, model learning, k-fold cross-validation, etc. The thesis will not delve into the details of the machine-learning algorithms. Any algorithm that supports numerical features can be applied to canonical representation, including SVMs, random forest, AdaBoost, decision trees, neural networks, multi-layer perceptrons, and others.

An important note considers k-fold cross-validation. Overlapping windows have a large degree of similarity and hence straight-forward k-fold cross-validation may produce an optimistic estimate of model performance. A better approach is to use folds that correspond to different sets of measurements or even different agents. For example, if the available dataset contains measurements of five agents, it makes sense to run *k-agent* cross-validation, where the model is trained on four agents and tested on the fifth. The procedure is repeated for each agent and results are averaged.

3.4 Reducing Spurious Activity Transitions

An activity-recognition model classifies each window into one of the predefined activities; however, regardless of our efforts, it may still mislabel activities. The model usually mislabels single moments or short intervals more often than longer intervals. Activity recognition can be improved by taking into account the continuity of activities, for example, the agent cannot switch between walking and lying every tenth of a second. Such transitions between activities that do not occur in reality, but are caused by mislabeling, are considered spurious.

One approach to enhancing the activity recognition model is to the extend feature vector with k previous activities as recognized by the classifier. This leads to a potential problem, namely, the machine-learning algorithm may learn that the current activity is always the same as the previous one, since this is often the case. The problem may be circumvented by having two activity recognition models θ_A and θ, where θ_A's attribute vector does not contain any previous activities, and θ's attribute vector contains k previous activities as recognized by θ_A. This way, even if θ heavily weights previous activities, those recognized by θ_A will change, as θ_A is not biased with θ's inertia.

Based on the above intuition, we introduce two candidate approaches for reducing spurious activity transitions: sequential grammar-based classifier (SGBS) and hidden Markov models (HMM). The input to these models is a sequence of actions $\mathbf{a}$ labeled by an activity recognition model θ_A.

Definition 6. Activity sequence $\mathbf{a}^{(T)}$ *is a totally-ordered sequence of* T *actions s.t.* $\mathbf{a}^{(T)} = \{a_t; 1 \leq t \leq T\}$.

The goal is to assign the best possible sequence of actions $\mathbf{a}'$ given a criterion and a sequence of input actions $\mathbf{a}$, often referred to as an observation sequence.

3.4.1 Sequential Grammar-Based Classifier

A sequential grammar-based classifier, introduced by Goshorn (2001), classifies an observed activity sequence $\mathbf{a}$ to the behavior which it most closely resembles. The similarity is defined in terms of the transformation cost of a sequence into a syntactically correct sequence belonging to that behavior.

Suppose there is a set of n possible behaviors $\mathbf{b}_i \in \mathbb{B}$, $1 \leq i \leq n$. A behavior $\mathbf{b}_i$ consists of a sequence of actions $a \in \mathbb{A}$ and is generated with a corresponding finite state machine $M_i(\mathbb{A}, S, s_o, \delta, F)$, where $\mathbb{A}$ is the input alphabet (activities), S is a finite, non-empty set of states, s_0 is an initial state, $\delta : S \times \mathbb{A} \rightarrow S$ is state-transition function, and F is the set of final states.

Suppose you want to classify an observation sequence $\mathbf{a}^{(k)} = \{a_1, a_2, ..., a_k\}$. If $\mathbf{a}^{(k)}$ is recognized by any of the automata M_i, then it is classified as behavior $\mathbf{b}_i$. If it is not, then it must be edited into a sequence that is. In order to do so, a new automaton M_i' is created, which is able to parse unrecognizable sequence $\mathbf{a}^{(k)}$ by transforming any symbol within its alphabet, but with an associated cost, as follows. There are two operations that are used for transformations: substitution and deletion. Let $S(a_i, a_j)$ denote the operation of substituting an input symbol a_i with a_j, and let $D(a_i)$ denote the operation of deletion of symbol a_i. Denote edit costs as $c_S(a_i, a_j)$ and delete cost $c_D(a_i)$, respectively.

In order to utilize the performance of the classifier θ_A, we use its performance measures obtained from its confusion matrix. The error rate for mislabeling an activity a_i with a_j is relevant for assigning the cost of substituting action a_i with a_j. Therefore, to derive $c_S(a_i, a_j)$, we simply invert the probability $\Pr\{a_i | a = a_j\}$ from the confusion matrix:

$$c_S(a_i, a_j) = \frac{1}{\Pr\{a_i | a = a_j\}}. \tag{3.20}$$

Similarly, the cost for deleting the action a_i is defined inversely proportional to the inherent probability that classifier θ_A labels a_i with true activity correctly. The deletion cost $c_D(a_i)$ is:

$$c_D(a_i) = \frac{1}{\Pr\{a_i | a = a_i\}}. \tag{3.21}$$

Suppose, that in order to transform the action sequence $\mathbf{a}^{(k)}$ into a behavior $\mathbf{b}_i$, there need to be $n_S(a_i, a_j)$ substitutions of a_i with a_j and $n_D(a_i)$ deletions of a_i. Then, the distance $d(\mathbf{a}^{(k)}, \mathbf{b}_i)$ between the action sequence $\mathbf{a}^{(k)}$ and a behavior $\mathbf{b}_i$ is given by Equation (3.22):

$$d(\mathbf{a}, \mathbf{b}_i) = \sum_{i=0}^{|\mathbb{A}|} \sum_{j=0}^{|\mathbb{A}|} c_S(a_i, a_j) n_S(a_i, a_j) + \sum_{i=0}^{|\mathbb{A}|} c_D(a_i) n_D(a_i). \tag{3.22}$$

The action sequence $\mathbf{a}^{(k)}$ is classified as a behavior $\mathbf{b}_i$ represented by a finite state machine M_i' that outputted the smallest distance:

$$\mathbf{a} = \arg\min_{\mathbf{b}_i \in \mathbb{B}} d(\mathbf{a}, \mathbf{b}_i). \tag{3.23}$$

In general, the lengths of behaviors $\mathbf{b}_i$ and action sequence $\mathbf{a}$ need not be be the same (although the lengths of behaviors $\mathbf{b}_i$ should be in to avoid length normalization). To avoid this problem, we use the overlapping sliding windows of length $|\mathbf{b}|$ and assign the classified behavior to the activity at the selected window index.

3.4.2 Hidden Markov Models

Hidden Markov model (HMM) is a temporal probabilistic model with two embedded stochastic processes: a hidden process Q that can be observed only through another visible process O. Each state has state-transition probabilities, which are visible, and a probability distribution over the possible values of $\mathbb{A}$. The key assumption is that the current hidden state of the agent is affected only by its previous state.

A hidden Markov model $\theta(\mathbb{H}, \mathbb{A}, \delta, \nu, \pi)$ is characterized by the following:

- $\mathbb{H} = \{h_i\}$ is a set of N hidden states, individual states are denoted as $\mathbb{H} = \{h_1, h_2, ..., h_N\}$, and the state at time t as Q_t,
- $\mathbb{A} = \{a_j\}$ is a set of distinct observation symbols (that is, activities) per state,
- $\delta = \{\delta_{ij}\}$ is the state transition probability distribution, where

$$\delta ij = \Pr\{q_{t+1} = s_j | q_t = h_i\}, 1 \leq i, j \leq N, \tag{3.24}$$

- $\nu = \{\nu_j(k)\}$ is the state observation probability distribution, where

$$\nu_j(k) = \Pr\{a_k | q_t = h_j\}, 1 \leq j \leq N, 1 \leq k \leq M, \tag{3.25}$$

- $\pi = \{\pi_i\}$ is the initial state distribution, where

$$\pi_i = \Pr\{q_1 = h_i\}, 1 \leq i \leq N. \tag{3.26}$$

An example of a hidden Markov model with $N = 4$ hidden states and $M = 3$ observation symbols is shown in Figure 3.10.

Given a learning set of observation sequences, there are two problems of interest that must be solved; that is, how to adjust model parameters so as to best describe agent behavior observation sequences, and, given a new observation sequence $\mathbf{a}^{(t)}$ and a model θ, how to choose a corresponding state sequence $\mathbf{s} = \{q_i = h_j | 1 \leq i \leq t, 1 \leq j \leq M\}$, that best explains the observations.

The first problem, where the goal is to adjust model parameters in order to maximize observation sequence probability, has no optimal solution. Parameters can be locally maximized with the Baum-Welch method (Baum et al., 1970), which maximizes the likelihood of

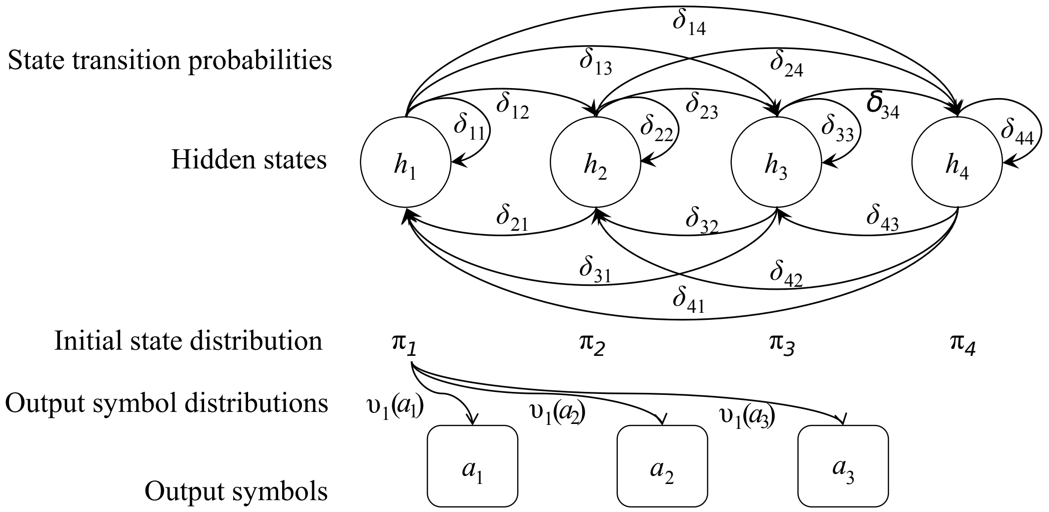


Figure 3.10: An example of a hidden Markov model with four hidden states and three observation symbols.

the training set. Instead of calculating the required state transitions from the observations, it iteratively estimates the parameters. The method starts with arbitrarily chosen values and then computes the expected frequencies by weighting the observed sequences over the current model probabilities. The expected frequencies substitute the old parameters and the procedure iterates until converging on a local maximum.

The second problem, *uncovering* the hidden part of the model, is solved with the Viterbi algorithm (Viterbi, 1967), which assumes that the output symbols in observation sequence $\mathbf{a}$ correspond to hidden state sequence $\mathbf{h}$. It finds the optimal state sequence $\mathbf{h}$ for the given observation sequence $\mathbf{a}$ in terms of maximizing the expected number of correct states, which is achieved with dynamic programming.

Suppose the task is to classify an observation sequence $\mathbf{a}$. First, an HMM model θ is created from the learning set of observations with the Baum-Welch method. Then, the Viterbi algorithm applied on model θ and observation sequence $\mathbf{a}$ returns the most probable hidden state sequence $\mathbf{h}$. In the last step, we take into account the θ's state observation probability distribution ν to assign the most probable symbol to each state; that is, it is assumed that a hidden state corresponds to the activity that is most likely observed in that state.

3.5 Compound-Activity Recognition

We use the term compound activity to describe behavior within an activity sequence. Activities, in particular atomic actions, are mainly used as behavior primitives describing the most elementary behavior aspects, while compound activities describe higher-level behavior aspects, usually spanned over longer periods of time.

Definition 7. Compound activity $b_k \in \mathbb{B}$ *describes an activity sequence* $\mathbf{a}^{(k)}$ *as behavior caused by an agent in a particular situation limited by time span* $1 \leq t \leq k$ *that explains activities* $a_1, ..., a_k$*, where* $\mathbb{A}$ *is a set of possible activities.*

The main difference between activities and compound activities is in the mapping function: an activity is assigned from observation vector(s), that is, $f : \mathbb{R}^{|\mathbf{x}|} \rightarrow \mathbb{A}$, while a compound activity is assigned from an activity sequence, that is, $g : \mathbb{A}^{|\mathbf{a}^{(k)}|} \rightarrow \mathbb{B}$. Given a set of possible compound activities $\mathbb{B} = \{b_i\}, 1 \leq i \leq K$, the goal is to classify an activity sequence $\mathbf{a}$ into one of the activities from $\mathbb{B}$.

For this task, we utilize hidden Markov models introduced in the previous section. The idea is to segment the learning set by different behavior types and to create an HMM model θ_i for each of the behaviors $b_i \in \mathbb{B}, 1 \leq i \leq K$.

Next, we compute the probability of the sequence $\mathbf{a}$ given a model θ_i, that is, $P[a|\theta_i]$. The straightforward approach through enumerating every possible state sequence of length $T = |\mathbf{a}|$ involves the order of $2TN^T$ calculations (Koller and Friedman, 2009), which is computationally unfeasible even for small values of N and T, for example, $N = 5$ states and $T = 100$ requires 10^{72} calculations. A more efficient procedure, denoted as the forward-backward algorithm (Rabiner, 1989), first computes a set of forward probabilities that predicts the likelihood of ending up in any particular state given the first t observations in the sequence $\mathbf{a}$. In the second pass, the algorithm computes a set of backward probabilities that predicts the likelihood of the remaining observations given any starting point t. These two probability distributions can then be combined to obtain the distribution over states at any specific point in time given the entire observation sequence. The reader is referred to Koller and Friedman (2009) for details.

Finally, the atomic action sequence $\mathbf{a}$ is classified as the behavior b_i that outputted the highest probability $\Pr\{\mathbf{a}|\theta_i\}$:

$$\arg\max_{b_i \in \mathbb{B}} \Pr\{\mathbf{a}|\theta_i\}. \tag{3.27}$$

3.6 Recognition of Agent-Agent Interactions

The previous sections were mainly focused on single-agent behavior. In order to assess all the behavior aspects, some domains require recognition of interactions among agents.

Definition 8. Interaction *between agents* A *and* B *or interactive behavior* $\chi_{i,j}(\langle \mathbf{a}_A, \mathbf{a}_B \rangle) \in \mathbb{I}$ *is behavior in time span* $i \leq t \leq j$ *that explains activity sequences* $\mathbf{a}_A$ *and* $\mathbf{a}_B$ *that correspond to activity sequence of the agent* A *and agent* B*, respectively.* $\mathbb{I}$ *is a set of possible interactions.*

In other words, an interaction describes joint behavior of two agents in a specific time span.

This section demonstrates an approach for interaction recognition based on coupled hidden Markov models (CHMMs), which are briefly described below. The reader is referred to Brand et al. (1997) for details. The observation sequence $\hat{\mathbf{a}} = \langle \mathbf{a}_A, \mathbf{a}_B \rangle$ consists of two activity sequences, namely $\mathbf{a}_A$ of agent A and $\mathbf{a}_B$ of agent B, when they are within some predefined radius R. The CHMMs are able to model complex, interactive behavior by two

HMM chains, where the hidden states from one chain directly impact the hidden states from the other chain.

Figure 3.11 illustrates a CHMM for a pair of action traces with length $l = 3$. The current state Q_t^A of agent A is affected by both its previous state Q_{t-1}^A and previous state Q_{t-1}^B of the agent B (similarly Q_t^B is affected by Q_{t-1}^B and Q_{t-1}^A). Each state Q_i also impacts the corresponding observation state Y_t. For example, if agent A moves toward agent B, the next state of the latter takes this into account and produces a corresponding atomic action, for example, an avoidance maneuver.

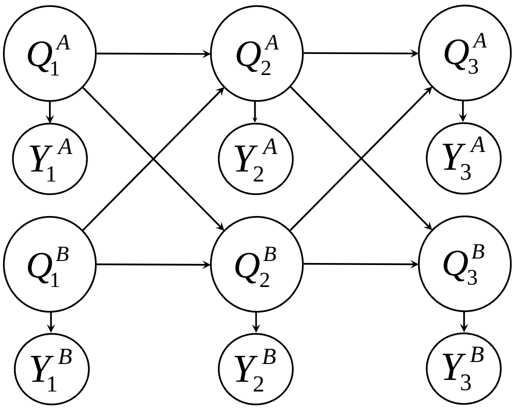


Figure 3.11: An example of CHMM for a pair of action traces with length $T = 3$.

Similarly to the previous section, we create a CHMM model $\hat{\theta}_i$ for each interaction χ_i from the set of possible interactions $\mathbb{I}$. For an observation sequence $\hat{\mathbf{a}}$, the posterior probability is computed given the model $\hat{\theta}_i$ using slightly modified standard HMM algorithms. The reader is referred to Brand et al. (1997) and Koller and Friedman (2009) for details. Finally, the interaction is classified by the model that outputs the highest probability $\Pr\{\hat{\mathbf{a}}|\hat{\theta}_i\}$:

$$\arg\max_{\chi_i \in \mathbb{I}} \Pr\{\hat{\mathbf{a}}|\hat{\theta}_i\}. \tag{3.28}$$

3.7 Summary and Discussion

This chapter addressed the activity recognition from sensor data, where considerable amount of noise is present. We introduced a pipeline-based approach, ARPipe, that includes noise removal, feature vector construction, activity recognition classifier, and spurious activity transition removal. The noise removal as well as feature vector construction steps were demonstrated on location-based sensors, which provide significantly less accurate measurements compared to location sensors used in related work (Sukthankar and Sycara, 2005; Qian et al., 2004). Since real locations, that is, true locations of a moving object, were infeasible to obtain, the noise removal was evaluated indirectly in (Kaluža and Dovgan, 2009; Luštrek et al., 2009), where it proved beneficial. The removal of spurious activity transitions was investigated in (Kaluža, 2009), where HMM approach achieved better results. In summary, the main two novelties presented in this chapter are activity recognition from noisy location sensors, and the ARPipe, which represents a comprehensive approach to activity recognition.

ARPipe provides the fist step to anomalous and suspicious behavior detection by recognizing behavior primitives, that is, activities. Additional approaches, such as compound activity recognition and recognition of agent interactions, help us to recognize more complex behaviors. The next chapter will discuss how to encode and evaluate such behavior components.

4 Behavior Signatures

This chapter discusses how to encode a sequence of actions into a behavior signature. It introduces a novel presentation denoted as a spatio-activity matrix, demonstrates a visualization technique, and proposes a feature extraction approach.

4.1 Definitions

In Chapter 3, we transformed a sequence of observation vectors to a higher-level description of behavior primitives. This chapter discusses how to efficiently encode the sequence of behavior primitives in order to perform additional analysis and effectively visualize the data. By effective visualization, we aim at a presentation that allows humans to quickly compare various behavior patterns and to find the main differences among them.

First, we define static points in the environment that refer to significant locations, specific spatial areas, or regions of partitioned space (for example, squared partitioning).

Definition 9. Static landmark s *is a point in the environment that remains fixed over the observed period of time. A set of static landmarks in the environment is denoted as* $\mathbb{S} = \{s_i\}$.

Next, given a set of static landmarks $\mathbb{S}$ and a set of activities $\mathbb{A}$, the behavior can be described in landmark-activity state space $\mathbb{S} \times \mathbb{A}$. The behavior can be then represented as a trajectory through the landmark-action state space as follows.

Definition 10. Behavior trace $\mathbf{b} = \{\langle a, s\rangle_t\}, 1 \leq t \leq T$ *is a sequence of tuples in which each tuple* $\langle a, s\rangle_t$ *indicates the environmental state* s *and the activity* a *being performed at this state at time step* t.

4.2 Spatio-Activity Matrix

This section introduces a behavior-trace encoding that captures activity distributions, landmark distribution, distribution of activities over landmarks, and distribution of landmarks over activities.

Consider a set $\mathbb{A} = \{a_1, a_2, ..., a_M\}$ of predefined activities and a set $\mathbb{S} = \{s_1, s_2, ..., s_K\}$ of static landmarks where the agent can be present. Let $\mathbf{v}_t$ denote a spatio-activity vector of size $M + K$ at time step t. The first M elements correspond to M activities in $\mathbb{A}$ and the last K elements correspond to static landmarks in $\mathbb{S}$; that is, i-th element corresponds to a_i if $i \leq M$ or s_{i-M} if $M < i \leq K$.

A tuple $(a, s)_t$ is then transformed to the spatio-activity vector with Equation (4.1), which assigns a binary value to i-th element of the vector: 1, if the activity is equal to i-th activity in $\mathbb{A}$ or if the landmark is equal to landmark at position $i - M$ in $\mathbb{S}$; 0, otherwise.

$$\mathbf{v}_{t(i)} = \begin{cases} 1 & ; \text{if } a = a_i, 1 < i \leq M \text{ or } s = s_{i-M}, M < i \leq K, \\ 0 & ; \text{otherwise.} \end{cases} \tag{4.1}$$

A spatio-activity vector is basically a binary representation of activities and landmarks present in a tuple.

Suppose we have two spatio-activity vectors $\mathbf{v}_i$ and $\mathbf{v}_j$. Let $\mathbf{t}(\mathbf{v}_i, \mathbf{v}_j)$ denote a transition vector from the spatio-activity vector $\mathbf{v}_i$ to $\mathbf{v}_j$ as an indication of a change constrained by $\|\mathbf{t}(\mathbf{v}_i, \mathbf{v}_j)\| = 1$:

$$\mathbf{t}(\mathbf{v}_i, \mathbf{v}_j) = \neg(\mathbf{v}_j \rightarrow \mathbf{v}_i), \tag{4.2}$$

where operator '$\rightarrow$' is binary implication and '$\neg$' is binary negation.

Now suppose we want to encode the behavior trace $\mathbf{b} = \{(a, s)_t\}, 1 \leq t \leq T$. First, we assign a new vector $\mathbf{v}_t$ to each tuple $(a, s)_t$. Let $\mathbf{M}(\mathbf{b})$ denote the spatio-activity matrix, where the dynamics of a person in the given behavior trace $\mathbf{b}$ is captured:

$$\mathbf{M}(\mathbf{b}) = \mathbf{v}_1\mathbf{v}_1{}^\intercal + \sum_{t\in[2,\ldots,T]} [\mathbf{v}_t\mathbf{v}_t{}^\intercal + \mathbf{t}(\mathbf{v}_{t-1}, \mathbf{v}_t)\mathbf{t}(\mathbf{v}_{t-1}, \mathbf{v}_t)^\intercal]. \tag{4.3}$$

At this step, the spatio-activity matrix $\mathbf{M}$ registered element frequency in a spatio-activity vector and their transitions. In order to make $\mathbf{M}$ comparable to other matrices constructed from behavior traces of different lengths, the matrix $\mathbf{M}$ must be normalized.

Define $norm(\mathbf{M})$ as an operation that normalizes the values of the matrix $\mathbf{M}$ to the interval $[0, 1]$. The $norm(\mathbf{M})$ is defined for an element $\mathbf{M}_{i,j} \in \mathbf{M}$ by the expression

$$\mathbf{M}_{i,j} = \begin{cases} \frac{\mathbf{M}_{i,j}}{\sum_{k=1}^{M} \mathbf{M}_{k,k}} & ; i = j \wedge i \leq M \\ \frac{\mathbf{M}_{i,j}}{\sum_{k=M+1}^{M+K} \mathbf{M}_{k,k}} & ; i = j \wedge i > M \\ \frac{\mathbf{M}_{i,j}}{\sum_{\substack{k=1 \\ l=1 \\ l\neq k}}^{M} \mathbf{M}_{k,l}} & ; i \neq j \wedge i \leq M \wedge j \leq M \\ \frac{\mathbf{M}_{i,j}}{\sum_{\substack{k=M+1 \\ l=M+1 \\ l\neq k}}^{M+K} \mathbf{M}_{k,l}} & ; i \neq j \wedge i > M \wedge j > M \\ \frac{\mathbf{M}_{i,j}}{\sum_{k=M+1}^{M+K} \mathbf{M}_{i,k}} & ; i \leq M \wedge j > M \\ \frac{\mathbf{M}_{i,j}}{\sum_{k=1}^{M} \mathbf{M}_{i,k}} & ; i > M \wedge j \leq M \end{cases}. \tag{4.4}$$

Intuitively, the matrix $\mathbf{M}$ consists of six regions as shown in Figure 4.1. The interpretation of the regions is as follows: the activity-activity part (top left) includes the fractions of the time spent performing particular activities (diagonal elements) and the distribution of transitions between activities (non-diagonal elements); the spatio-spatio part (bottom right) includes the fractions of time spent at the particular landmarks (diagonal elements) and the distribution of transitions between landmarks (non-diagonal elements); the activity-spatio part (top right) describes the distribution of activities over landmarks; and the spatio-activity part (bottom left) describes the distribution of landmarks over activities.

The complete spatial-activity construction procedure is described in Algorithm 4.1. The input is a behavior trace $\mathbf{B}$. Each tuple $(a, s)_t \in \mathbf{B}$ is first transformed into the spatio-activity vector $\mathbf{v}_t$ using Equation (4.1) and added to a temporary sequence of vectors $\mathbf{V}$. The sequence V is then used to compute the spatio-activity matrix $\mathbf{M}$ using Equation (4.3). Finally, the matrix $\mathbf{M}$ is normalized by Equation (4.4).

4.2.1 Time Complexity Analysis

The runtime complexity increases linearly with the behavior trace length T and quadratically with the sizes of sets $\mathbb{A}$ and $\mathbb{S}$. First, $T(M + K)$ operations are required to transform each tuple to a spatio-activity vector. Next, there are $2(T - 1)(M + K)$ operations to

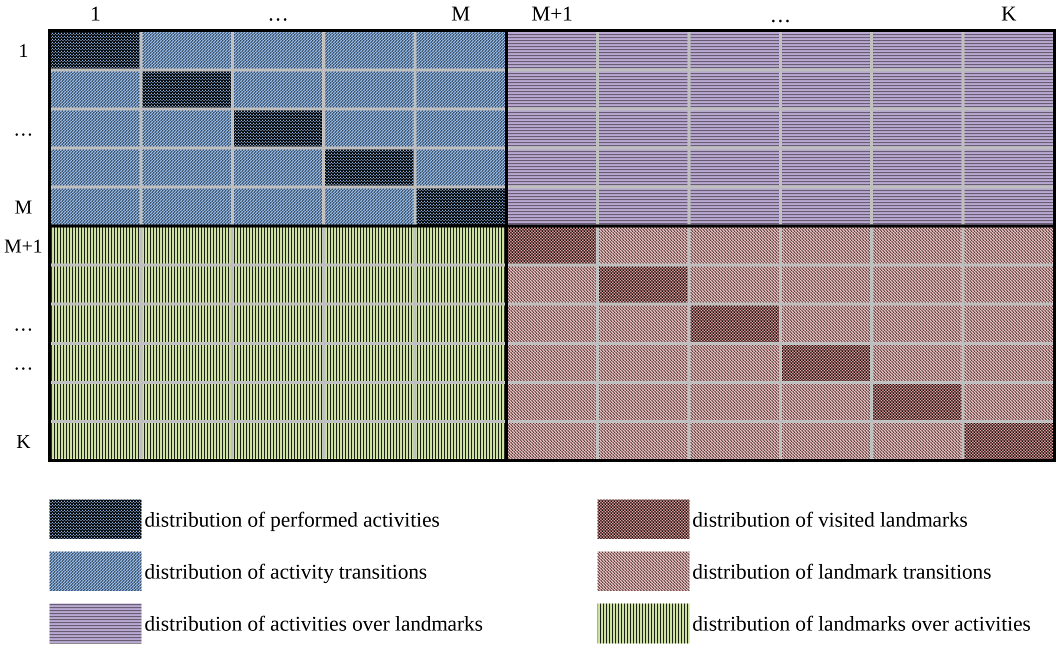

1
...
M
M+1
...
K
1
...
M
M+1
...
...
K
distribution of performed activities
distribution of activity transitions
distribution of activities over landmarks
distribution of visited landmarks
distribution of landmark transitions
distribution of landmarks over activities

Figure 4.1: Regions of spatio-activity matrix.

compute $T-1$ transition vectors using implication and negation on each pair of spatio-activity vectors. The next step builds the spatio-activity matrix, where each of vector-vector multiplications requires $2(M+K)$ operations that result in matrices. Matrix summarization requires $(M+K)^2$ steps, and is applied for each T. In total there are $2(M+K)+(M+K)^2+T(4(M+K)+(M+K)^2)$ operations. The normalization step can be implemented by pre-computing the six divisors and applying them on each element in the matrix. This requires $T(M+K+M^2+K^2+2(MK))$ operations in total. The overall time complexity, then, is $O(T(M+K)^2)$.

In practice, however, the behavior trace is a few orders of magnitude larger than the number of activities and landmarks; that is, $(M+K) << T$, which makes the computation of spatio-activity matrix dependent only on the behavior trace length T.

4.2.2 Visualizations

Since the matrix $\mathbf{M}$ is normalized to the interval $[0,1]$, it can be visualized with a color map. Figure 4.2 shows an example of such a matrix $\mathbf{M}$ visualization, where the color ranges from low frequency (blue) to high frequency (red); a warmer color represents a higher intensity (see the legend on the left-hand side). It shows a behavior matrix constructed from a daily behavior trace of a person for activities $\mathbb{A}=\{lying, sitting, standing\}$ and landmarks $\mathbb{S}=\{lounge, bedroom, kitchen, toilet\}$.

Figure 4.2 can be interpreted as follows. There are three red squares that indicate a high ratio: $M_{bedroom,lying}$ indicates that lying is the prevailing bedroom activity; $M_{kitchen,standing}$ indicates that standing is the prevailing kitchen activity; and $M_{lying,bedroom}$ indicates that lying is mostly carried out in the bedroom. Next, blue squares indicate activity absence; for

Algorithm 4.1 Create spatio-activity matrix.

```
Require: behavior trace $\mathbf{b} = \{\langle a,s\rangle_1, \langle a,s\rangle_2, ..., \langle a,s\rangle_n\}$
Ensure: normalized matrix $\mathbf{M}(\mathbf{b})$
  $\mathbf{V} \leftarrow \{\}$
  for $\langle a,s\rangle \in \mathbf{b}$ do
    $\mathbf{v} \leftarrow sa\_vector(\langle a,s\rangle)$
    $\mathbf{V} \leftarrow \mathbf{V} \cup \mathbf{v}$
  end for
  $\mathbf{M} \leftarrow \mathbf{v}_1 \mathbf{v}_1^{\intercal}$
  for $v_i \in \mathbf{V}$, $i > 1$ do
    $\mathbf{M} \leftarrow \mathbf{M} + \mathbf{v}_i \mathbf{v}_i^{\intercal} + \mathbf{t}_{i-1,i} \mathbf{t}_{i,i-1}^{\intercal}$
  end for
  $norm(\mathbf{M})$
```

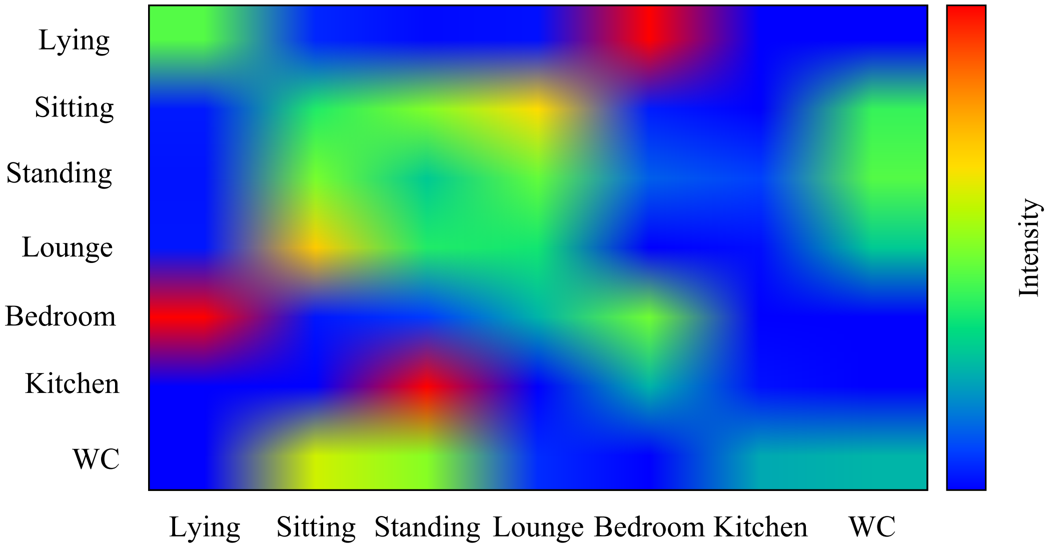


Figure 4.2: Visualization of a daily spatio-activity matrix of one person. A warmer color represents a higher value.

example, $M_{kitchen,lying}$ and $M_{kitchen,sitting}$ indicate that lying and sitting rarely occurred in the kitchen. Diagonal elements in the bottom-right part of the table (that is, $M_{lounge,lounge}$, $M_{bedroom,bedroom}$, $M_{kitchen,kitchen}$, $M_{toilet,toilet}$) reveal that the person spent most of the time in the bedroom followed by the lounge and toilet, and almost no time in the kitchen.

The visualization is especially useful in a comparison of multiple behavior traces. A small change, for example, in the ratio between sleeping in the bed (being ill) and walking around the apartment (a healthy person), is rapidly propagated through the spatio-activity matrix and, therefore, one can quickly notice the change and the type of change at the same time.

4.2.3 Feature Extraction

The behavior matrix can be directly fed into models for anomalous and suspicious behavior detection (discussed in Chapter 5) by transforming the spatio-activity matrix $\mathbf{M}^{M+K,M+K}$ to a feature vector $\mathbf{m}$. Note that the vector $\mathbf{m}$ contains $(M+K)^2$ elements, and some of them rarely change in different behavior traces. The main idea, then, is to reduce the number of features to the most representative set. This section presents an approach based on dimensionality reduction technique denoted as principal component analysis.

Principal component analysis (PCA) is an orthogonal linear transformation of possibly correlated variables onto a subspace. The choice of the k-dimensional projection subspace is made such that the projection distances have a minimal deformation: squares of the k-dimensional subspace distances are as large as possible. By projecting the data onto the new coordinate system, the greatest variance emerges on the first coordinate (called the first principal component), while the second greatest variance emerges on the second coordinate, and so on.

Implementing PCA is the equivalent of applying Singular Value Decomposition (SVD) to the covariance matrix. Consider a set of spatio-activity matrices $\mathbb{M} = \{\mathbf{M}_i\}$, $1 \leq i \leq L$. Each spatio-activity matrix $\mathbf{M}_i$ is unrolled into a vector $\mathbf{m}_i$. Then we construct a matrix $\widehat{\mathbf{M}}$, which consists of L vectors $\mathbf{m}_i$, each unrolled from $\mathbf{M}_i$, $i = 1...L$.

$$\widehat{\mathbf{M}} = \begin{bmatrix} \mathbf{m}_1 \\ \mathbf{m}_2 \\ \vdots \\ \mathbf{m}_L \end{bmatrix} = \begin{bmatrix} \mathbf{M}_{1,1} & \mathbf{M}_{1,2} & \cdots & \mathbf{M}_{1,(M+K)^2} \\ \mathbf{M}_{2,1} & \mathbf{M}_{2,2} & \cdots & \mathbf{M}_{2,(M+K)^2} \\ \vdots & \vdots & \ddots & \cdots \\ \mathbf{M}_{L,1} & \mathbf{M}_{L,2} & \cdots & \mathbf{M}_{L,(M+K)^2} \end{bmatrix} \tag{4.5}$$

The PCA proceeds as follows: first, we compute the mean vector $\bar{\mu}$ with μ_j elements, $1 \leq j \leq (M+N)^2$ with Equation (4.6), and subtract the mean from $\widehat{\mathbf{M}}$ with Equation (4.7), which gives us matrix $\widehat{\mathbf{M}}_z$ with zero mean.

$$\mu_j = \frac{1}{L}\sum_{k=1}^{L} \mathbf{M}_{k,j}, 1 \leq j \leq (M+N)^2 \tag{4.6}$$

$$\widehat{\mathbf{M}}_z = \widehat{\mathbf{M}} - \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix}_L \bar{\mu} \tag{4.7}$$

Next, a matrix $\mathbf{C}$ of variances and covariances is computed with Equation (4.8), where the diagonal elements $i = j$ are variances σ_{ij}^2 and the non-diagonal elements $i \neq j$ are covariances $\sigma_i \sigma_j$.

$$\mathbf{C} = \frac{1}{L}\widehat{\mathbf{M}}_z \widehat{\mathbf{M}}_z^{\intercal} \tag{4.8}$$

Finally, $\mathbf{C}$ is decomposed into three matrices with SVD (Equation (4.9)). $\mathbf{S}$ is a diagonal matrix that stores singular values $\lambda_1, \lambda_2, ..., \lambda_n$. $\mathbf{U}$ and $\mathbf{V}$ are orthogonal matrices, while their column vectors are the so-called left and right eigenvectors of $\mathbf{C}$.

$$\mathbf{C} = \mathbf{U}\mathbf{S}\mathbf{V}^{\intercal} \tag{4.9}$$

When these eigenvectors multiply $\widehat{\mathbf{M}}_z$, the coordinates are shifted and rotated until they end up aligned with basis vectors. Note that PCA now re-expresses the data as a linear combination of its basis vectors, $\widehat{\mathbf{M}}_z\mathbf{V}$. $\mathbf{V}$ columns are found to produce the desired linear combinations. The first $\mathbf{V}$ column corresponds to the largest principal component, the second column corresponds to the second largest, and so on. These define the direction in which the variability of the original data set is maximized.

The transformed data now enable the use of anomalous and suspicious behavior detection models. All the behavior matrices $\{\mathbf{M}_i\} \in \mathbb{M}$, $1 \leq i \leq L$ are first expressed in the new coordinate system. When a new behavior matrix $\mathbf{M}_{L+1}$ is obtained, it is first expressed in the same system; the first components are subsequently used for further detection.

4.3 Summary and Discussion

This chapter proposed a novel, efficient encoding denoted as spatio-activity matrix that is able to capture behavior dynamics in a specific time period using spatio-temporal features. We provided a visualization technique to compare different behavior patterns. We further provided a feature extraction technique based on principal component analysis to reduce the spatio-activity matrix dimensionality, which can be directly used in anomaly detection algorithms. Compared to related work based on HMMs (Monekosso and Remagnino, 2010), the behavior patterns dynamics is expressed explicitly and can be visualized. Moreover, in contrast to research based on rule induction (Lee et al., 2004; Lymberopoulos et al., 2008), the presentation does not extract the exact behavior patterns, which leads to better generalization.

The obtained spatio-temporal features or their principal componentes will be used in the next chapter, which evaluates behavior patterns to perform anomalous and suspicious behavior detection. The spatio-temporal matrix can be constructed on various time intervals, such as hours, days, weeks, which provides different behavior encoding granularities; Section 5.4 discusses how to combine such evaluations. Also, the approach was demonstrated on spatio-temporal feature space, but in general, it can be applied on other feature spaces as well, which is an interesting direction for further investigations.

5 Anomalous and Suspicious Behavior Detection

This chapter discusses how to approach anomalous and suspicious behavior detection. It first formalizes the problem and shows how to optimally perform detection. It then discusses why optimal detection is not always possible, proves the lower error bound and discusses heuristics approaches. Finally, it describes how to design multi-view detectors and combine their evaluations.

5.1 Detection Objectives

Our focus is detection of deviant behavior patterns that might represent a security risk, health problem, or any other abnormal behavior contingency. Such patterns occur infrequently; however, when they do occur, their consequences can be quite dramatic, and often negatively. Typical examples include credit card fraud detection, cyber intrusions, industrial damage, etc.

More formally, we define behavior pattern as follows:

Definition 11. Behavior pattern $\tilde{\mathbf{b}}$ *is a vector of features extracted from behavior trace* $\mathbf{b} = \{(a, s)_t | 1 \leq t \leq T\}$*.*

The definition implies that a behavior pattern can be constructed from a behavior trace by an arbitrary function. The main idea is to introduce a set of features that effectively encodes the behavior trace, such as the spatio-activity matrix introduced in Chapter 4.

We use term *deviant behavior* to refer to behavior that is either suspicious or anomalous and cast the deviant behavior detection problem in a statistical framework, which builds upon Helman and Liepins (1993) intrusion detection framework. This will help introduce rigorous notions, which are required later in the thesis, and allow future work to evolve toward broader objectives.

At time step t, we observe a behavior pattern $\tilde{\mathbf{b}}_t$, generated by a hidden stochastic process H. Now suppose that H is a mixture of two auxiliary stochastic processes, namely the normal process N and the suspicious process S, that correspond to a legitimate and a deviant behavior of an agent, respectively. The random variable $y_t = 0$ if $\tilde{\mathbf{b}}_t$ is generated by N and $y_t = 1$ if $\tilde{\mathbf{b}}_t$ is generated by S. In reality, there can be many subprocesses contributing to both N and S; that is, many legitimate agents with different behavior patterns or an agent with many legitimate and deviant behavior profiles. However, here we assume only a single N and a single S that capture all the variability.

To this point, we have assumed that an observer is able to observe perfectly whether a behavior pattern is generated by S or N. In practice, however, it may appear that a legitimate agent emits deviant behavior patterns (or vice-versa). An observer might be limited for various reasons, such as an inability to detect characteristic features, and noisy activity recognition models. Therefore, we relax this assumption as follows: a behavior pattern $\tilde{\mathbf{b}}_t$ is observed as generated by N with the probability

$$n(\tilde{\mathbf{b}}_t) = \Pr\{H(t) = \tilde{\mathbf{b}}_t | y_t = 0\}, \tag{5.1}$$

and as generated by S with the probability

$$s(\tilde{\mathbf{b}}_t) = \Pr\{H(t) = \tilde{\mathbf{b}}_t | y_t = 1\}. \tag{5.2}$$

Given the preceding assumption, that is, *a priori* probability λ that a behavior pattern is legitimate, the mixture distribution of a pattern $\tilde{\mathbf{b}}_t$ is

$$\Pr\{H(t) = \tilde{\mathbf{b}}_t\} = \lambda n(\tilde{\mathbf{b}}_t) + (1-\lambda)s(\tilde{\mathbf{b}}_t). \tag{5.3}$$

Note that in most applications, λ is close to 1, since deviant behavior patterns are rare, necessitating the application of modeling techniques such as those described in the next section.

We illustrate via a simple example the above-mentioned definitions and concepts. Suppose the behavior pattern consists of only two features, *activity* and *landmark*, and that the possible values for *activity* are $\mathbb{A} = \{standing, lying\}$, while the possible values for *landmark* are $\mathbb{S} = \{kitchen, bedroom\}$. Hence, the spatio-activity space can be presented by the set $\{\langle kitchen, standing\rangle, \langle kitchen, lying\rangle, \langle bedroom, standing\rangle, \langle bedroom, lying\rangle\}$ of ordered pairs. Assume the following probability distribution on spatio-activity space for $1 \leq t \leq T$:

$$\begin{aligned}
\Pr\{H(t) = \langle kitchen, standing\rangle | y_t = 0\} = n(\langle kitchen, standing\rangle) &= 0.950,\\
\Pr\{H(t) = \langle kitchen, lying\rangle | y_t = 0\} = n(\langle kitchen, lying\rangle) &= 0.020,\\
\Pr\{H(t) = \langle bedroom, standing\rangle | y_t = 0\} = n(\langle bedroom, standing\rangle) &= 0.250,\\
\Pr\{H(t) = \langle bedroom, lying\rangle | y_t = 0\} = n(\langle bedroom, lying\rangle) &= 0.850,\\
\Pr\{H(t) = \langle kitchen, standing\rangle | y_t = 1\} = s(\langle kitchen, standing\rangle) &= 0.020,\\
\Pr\{H(t) = \langle kitchen, lying\rangle | y_t = 1\} = s(\langle kitchen, lying\rangle) &= 0.900,\\
\Pr\{H(t) = \langle bedroom, standing\rangle | y_t = 1\} = s(\langle bedroom, standing\rangle) &= 0.050,\\
\Pr\{H(t) = \langle bedroom, lying\rangle | y_t = 1\} = s(\langle bedroom, lying\rangle) &= 0.050.
\end{aligned}$$

Then, if, for example, $\lambda = 0.9$, the mixture probability is:

$$\begin{aligned}
\Pr\{H(t) = \langle kitchen, standing\rangle\} &= 0.857,\\
\Pr\{H(t) = \langle kitchen, lying\rangle\} &= 0.108,\\
\Pr\{H(t) = \langle bedroom, standing\rangle\} &= 0.230,\\
\Pr\{H(t) = \langle bedroom, lying\rangle\} &= 0.770.
\end{aligned}$$

The objective of behavior detection is to identify those patterns that are likely to be deviant activities, that is, patterns $\tilde{\mathbf{b}}$ for which

$$\Pr\{y_t = 1 | H(t) = \tilde{\mathbf{b}}_t\} > \tau, \tag{5.4}$$

is above some threshold τ or is large relative to the probability for other traces.

According to Bayes theorem and our definitions

$$\begin{aligned}
\Pr\{y_t = 1 | H(t) = \tilde{\mathbf{b}}_t\} = &\\
&= \frac{(1-\lambda)\Pr\{H(t) = \tilde{\mathbf{b}}_t | y_t = 1\}}{(1-\lambda)\Pr\{H(t) = \tilde{\mathbf{b}}_t | y_t = 1\} + \lambda \Pr\{H(t) = \tilde{\mathbf{b}}_t | y_t = 0\}}\\
&= \frac{(1-\lambda)s(\tilde{\mathbf{b}})}{(1-\lambda)s(\tilde{\mathbf{b}}) + \lambda n(\tilde{\mathbf{b}})}\\
&= \frac{(1-\lambda)r(\tilde{\mathbf{b}})}{(1-\lambda)r(\tilde{\mathbf{b}}) + \lambda}\\
&= \frac{r(\tilde{\mathbf{b}})}{r(\tilde{\mathbf{b}}) + \lambda/(1-\lambda)},
\end{aligned} \tag{5.5}$$

where $r(\tilde{\mathbf{b}}) = s(\tilde{\mathbf{b}})/n(\tilde{\mathbf{b}})$ if $n(\tilde{\mathbf{b}}) > 0$, and $r(\tilde{\mathbf{b}}) = \infty$ if $n(\tilde{\mathbf{b}}) = 0$. We derive from Equation (5.4) and Equation (5.5) that a behavior pattern is deviant iff:

$$r(\tilde{\mathbf{b}}) > \frac{\lambda\tau}{(1-\lambda)(1-\tau)}. \tag{5.6}$$

Assuming the probabilities in the previous example, we can compute

$$\begin{aligned} r(\langle \mathit{kitchen}, \mathit{standing}\rangle) &= 0.021, \\ r(\langle \mathit{kitchen}, \mathit{lying}\rangle) &= 0.450, \\ r(\langle \mathit{bedroom}, \mathit{standing}\rangle) &= 0.200, \\ r(\langle \mathit{bedroom}, \mathit{lying}\rangle) &= 0.059. \end{aligned}$$

However, in practice, some or all of the probabilities for the above calculations are unknown. We have no direct knowledge of the *a priori* probability λ and distribution of processes N and S. Our primary concern is hence to develop models that estimate the likelihood of ratio $r(\tilde{\mathbf{b}})$ and, therefore, distributions $s(\tilde{\mathbf{b}})$ and $n(\tilde{\mathbf{b}})$.

5.2 Detection Performance

First, we define *detector* as a behavior signature ranking mechanism. The larger the value provided by the detector, the more deviation is attributed to the signature.

Definition 12. Graded detector D_g *is a function from behavior signature space* $\tilde{\mathbb{B}}$ *to non-negative real set:*

$$D_g : \tilde{\mathbb{B}} \to \mathbb{R}_0^+.$$

Definition 13. Binary detector D_g *is a function from behavior signature space* $\tilde{\mathbb{B}}$ *to binary set:*

$$D_b : \tilde{\mathbb{B}} \to \{0, 1\}.$$

For any graded detector D_g, we can associate a set of binary detectors $D_{b,\tau}$ satisfying

$$D_{b,\tau} = \begin{cases} 1; & D_g \geq \tau \\ 0; & \text{else} \end{cases}. \tag{5.7}$$

For example, we could define a graded detector as $D_g(\tilde{\mathbf{b}}) = r(\tilde{\mathbf{b}})$. Similarly, we could define a binary detector $D_b(\tilde{\mathbf{b}}) = 1$ if $r(\tilde{\mathbf{b}}) \geq \lambda/(1-\lambda)$ and $D_b(\tilde{\mathbf{b}}) = 0$ otherwise.

In general, legitimate process N and deviant process S may overlap, which means that for some signatures $\tilde{\mathbf{b}}$, both $n(\tilde{\mathbf{b}})$ and $s(\tilde{\mathbf{b}})$ are non-zero. Therefore, a detector (graded or binary) can map two signatures to the same value, even if one was generated by N and the other by S. Consequently, some error is unavoidable.

There are four possible outcomes as shown in Table 5.1. The first column contains possible signatures (deviant or legitimate), while the first line contains detector outcomes. There are two types of error: type I error or false positive, which can be thought of as *convicting a legitimate agent*; and type II error or false negative; that is, *letting a suspicious agent go free.* A perfect detector would have 0% false negatives and 0% false positives; that is, it would correctly identify all deviant and legitimate signatures. However, theoretically any detector will possess a minimum error bound if the distributions N and S overlap.

Figure 5.1 illustrates an example of overlapping distributions N and S (Duda et al., 2000). The red and blue shaded areas in the tail of distributions N and S represent type

Table 5.1: Detection outcomes.

	Detector: legitimate	Detector: deviant
Deviant signature	miss (false negative)	hit (true positive)
Legitimate signature	correct rejection (true negative)	false alarm (false positive)

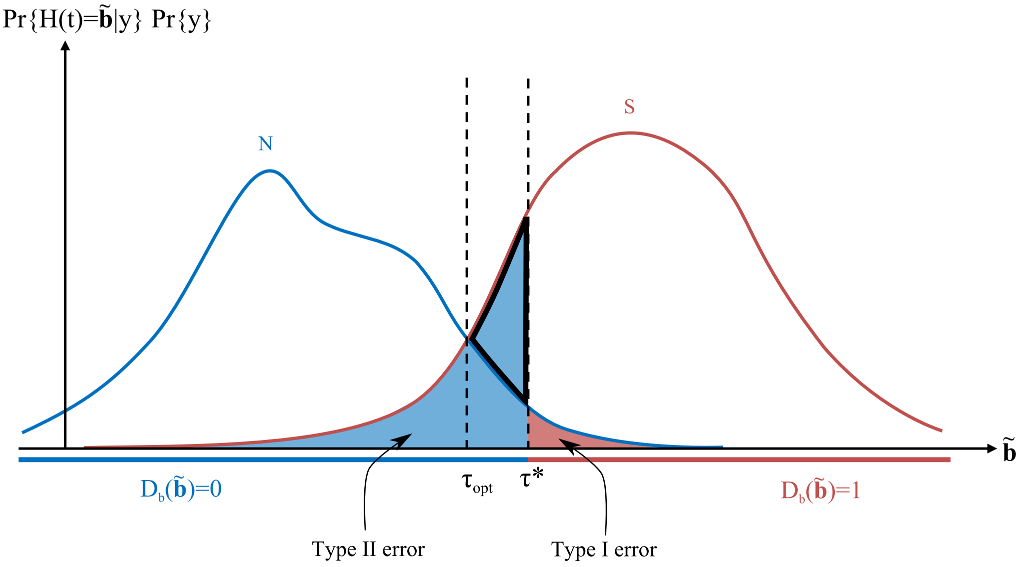


Figure 5.1: Overlapping distributions N and S.

I and type II errors, respectively. An arbitrarily chosen decision point $\tau*$ represents the binary detector's decision boundary; that is, regions $D_b(\tilde{\mathbf{b}}) = 0$ and $D_b(\tilde{\mathbf{b}}) = 1$. The marked triangular area represents *reducible error*, which can be eliminated if the decision boundary is moved to τ_{opt}, the optimal decision boundary, which gives the lowest error probability.

Theorem 1. *For any binary detector D_b, the symmetric error is bounded below by*

$$\sum_{t=1}^{T} \min\left(s(\tilde{\mathbf{b}}_t)(1-\lambda), n(\tilde{\mathbf{b}}_t)\lambda\right).$$

Proof. The error probability consists of two terms; that is, type I and type II errors as shown in Equation (5.8).

$$\begin{aligned}\Pr\{error\} &= \sum_{\substack{t=1\\ D_b(\tilde{\mathbf{b}})=0}}^{T} s(\tilde{\mathbf{b}}_t)\Pr\{y=1\} + \sum_{\substack{t=1\\ D_b(\tilde{\mathbf{b}})=1}}^{T} n(\tilde{\mathbf{b}}_t)\Pr\{y=0\}\\ &= \sum_{\substack{t=1\\ D_b(\tilde{\mathbf{b}})=0}}^{T} s(\tilde{\mathbf{b}}_t)(1-\lambda) + \sum_{\substack{t=1\\ D_b(\tilde{\mathbf{b}})=1}}^{T} n(\tilde{\mathbf{b}}_t)\lambda\end{aligned} \tag{5.8}$$

As follows from Equation (5.8), it is advantageous to classify a behavior signature $\tilde{\mathbf{b}}$ as legitimate if $s(\tilde{\mathbf{b}}_t)(1-\lambda) < n(\tilde{\mathbf{b}}_t)\lambda$ so that the smaller quantity will contribute to the error sum; the other way around follows analogously. Consequently, the lower error bound as defined in the theorem's statement is achieved. □

Frequently, the type I and type II errors are not considered of equal importance; hence, we can weight them with two constants $\alpha, \beta \in [0,1]$ s.t. $\alpha + \beta = 1$. The error probability then follows from Equation (5.9).

$$\Pr\{error\} = \alpha \sum_{\substack{t=1 \\ D_b(\tilde{\mathbf{b}})=0}}^{T} s(\tilde{\mathbf{b}}_t)(1-\lambda) + \beta \sum_{\substack{t=1 \\ D_b(\tilde{\mathbf{b}})=1}}^{T} n(\tilde{\mathbf{b}}_t)\lambda \tag{5.9}$$

5.2.1 Performance Measures

Optimality conditions assume that detection is performed with the benefit of perfect information. In practice, knowledge of distributions is not readily available and the detectors are evaluated on a behavior-signature database. The common performance measures are:

- *sensitivity*, also *true positive rate* or *recall* – the proportion of deviant signatures, which are correctly identified as such

$$\text{sensitivity} = \frac{\text{TP}}{\text{TP} + \text{FN}}, \tag{5.10}$$

- *specificity*, or *true negative rate* – the proportion of correctly identified legitimate signatures

$$\text{specificity} = \frac{\text{TN}}{\text{TN} + \text{FP}}, \tag{5.11}$$

- *precision*, or *positive predictive value* – the proportion of correctly raised alarms

$$\text{precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}. \tag{5.12}$$

In addition, sensitivity and precision, that is, the fraction of detected deviant signatures and the fraction of relevant alarms are ideally close to 100% (that is, no type I and type II errors). *F-measure* considers both the precision and the recall to compute the score as a weighted precision and recall average, where the score reaches its best value at 1 and worst at 0.

$$F = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}} \tag{5.13}$$

5.2.2 False Positive Paradox

The false positive paradox (Rheinfurth et al., 1998) occurs when the behavior signature-database has a low deviant behavior signature share and the detector's hit rate is lower than the false positive rate. In this case, the false positive test is more probable than the true positive test.

Consider the following example: suppose a detector D has false alarm rate 0.04% and false negative rate (miss) 0%; that is, it marks all deviant signatures. Now suppose we have a behavior signature database $\mathcal{B}_A$ with a million signatures in which 2 of 100 signatures are deviant; that is, 2%. The detection scores are as follows:

$$\text{TP} = 1{,}000{,}000 * 0.02 * 1.0 = 20{,}000 \text{ signatures (hits)},$$
$$\text{FP} = 1{,}000{,}000 * 0.98 * 0.0004 = 392 \text{ signatures (false alarms)}.$$

Hence, a signature is classified as deviant with over 98% confidence; that is, $precision = 20{,}000/20{,}392$.

Now suppose the same detection is applied to a behavior signature database $\mathcal{B}_B$ in which 1 of $10,000$ signatures are deviant; that is, 0.01%. The detector in this case scores as follows:

$$\begin{aligned} \text{TP} &= 100,000 * 0.0001 * 1.0 = 100 \text{ signatures (hits)}, \\ \text{FP} &= 100,000 * 0.9999 * 0.0004 \approx 400 \text{ signatures (false alarms)}. \end{aligned}$$

A signature is now classified as deviant with only 20% confidence, since only 100 of total 500 signatures marked as deviant are indeed deviant. Given the results on database $\mathcal{B}_A$, it is considered paradoxical that deviant detection is mostly a false alarm in database $\mathcal{B}_B$. The probability of a deviant detection result is hence determined not only by the accuracy of the detection, but also by the distribution of the sampled behavior signatures. Hence it is crucial that historical behavior signature distribution matches the behavior signature distribution on which the detector is applied.

5.3 Detectors

Helman and Liepins (1993) divided detectors into: *modeling approaches*, which require estimation of distributions $s(\tilde{\mathbf{b}})$ and $n(\tilde{\mathbf{b}})$ to directly attack the problem of deviant behavior detection, for example, neural networks (Biermann et al., 2001), k-nearest neighbors (Govindarajan and Chandrasekaran, 2009), naïve-Bayes estimators (Kruegel et al., 2003); and *non-modeling approaches* that do not explicitly estimate $s(\tilde{\mathbf{b}})$ or $n(\tilde{\mathbf{b}})$, but use various heuristics to flag deviant behavior patterns, for example, expert rules (Esponda et al., 2004), plan recognition (Avrahami-Zilberbrand and Kaminka, 2007), and decision trees (Ektefa et al., 2010).

5.3.1 Frequentist Estimator

Frequentist estimator is a basic approach that estimates distributions $n(\tilde{\mathbf{b}})$ or $s(\tilde{\mathbf{b}})$. The distribution $n(\tilde{\mathbf{b}})$ is estimated from a historical database of behavior signatures $\mathcal{B}$:

$$\bar{n}(\tilde{\mathbf{b}}) = \Pr\{\tilde{\mathbf{b}}\} = \frac{\eta(\tilde{\mathbf{b}})}{|\mathcal{B}|}, \tag{5.14}$$

where $\eta(\tilde{\mathbf{b}})$ counts the number of the occurrences of the signature $\tilde{\mathbf{b}}$ in $\mathcal{B}$.

Two promising models for estimating distribution $s(\tilde{\mathbf{b}})$ are *uniform* and *independence* model. The uniform model assumes that all signatures are equally likely, regardless of the historical database:

$$\bar{s}_u(\tilde{\mathbf{b}}) = \frac{1}{|\tilde{\mathbf{b}}|}. \tag{5.15}$$

The *independence* model assumes that the features in the behavior signature $\tilde{\mathbf{b}} = \langle f_1, ..., f_L\rangle$ are independent, that is:

$$\bar{s}_i(\tilde{\mathbf{b}}) = \Pr\{f_1, \ldots, f_L\} = \prod_{i=1}^{L} \Pr\{f_i\} = \prod_{i=1}^{L} \frac{\eta(f_i)}{|\mathcal{B}|}. \tag{5.16}$$

The models are best illustrated by means of example. Consider a behavior signature structure from the previous example. Suppose further that the historical database consists of 100 signatures, yielding observed frequencies of the four possible behavior signatures as shown in Table 5.2.

The frequentist estimator gives us, for example, $\bar{n}(\langle kitchen, standing\rangle) = 0.030$ and $\bar{n}(\langle kitchen, lying\rangle) = 0.070$. With the uniform model, we obtain $\bar{s}_u(\tilde{\mathbf{b}}) = 0.25$ for any

Table 5.2: Example: Observed frequencies of four possible behavior signatures.

Activity / Landmark	Kitchen	Bedroom
Standing	3	8
Lying	7	82

$\tilde{\mathbf{b}}$, which gives us ratios $\bar{r}_u(\langle kitchen, standing\rangle) = 8.333$ and $\bar{r}_u(\langle kitchen, lying\rangle) = 3.571$. The uniform model simply exposes signatures less likely to be generated by process N, without any reference to process S. Hence, the signature $\langle kitchen, standing\rangle$ is considered as anomalous, as it is the least frequent in the historical database.

The independence model gives us $\bar{s}_i(\langle kitchen, standing\rangle) = 0.011$ and $\bar{s}_i(\langle kitchen, lying\rangle) = 0.089$, leading us to ratios $\bar{r}_i(\langle kitchen, standing\rangle) = 0.367$ and $\bar{r}_i(\langle kitchen, lying\rangle) = 1.271$. In this case, the signature $\langle kitchen, lying\rangle$ is marked as suspicious because activity *lying* is something relatively unusual to be performed at landmark *kitchen*.

Each of the models is clearly better than the other for certain tasks. While we believe those basic models to be reasonable starting points, we envision a framework in which several models are combined to yield the final evaluation.

5.3.2 Density Estimator

The estimation can be based on a distance measure to deviant and legitimate behavior signatures in the historical database $\mathcal{B}$. Suppose the historical database $\mathcal{B}$ can be split into legitimate $\mathcal{B}_n$ and deviant $\mathcal{B}_s$ signatures. The ratio $r(\tilde{\mathbf{b}})$ can be then defined as a ratio between distance to the k-nearest legitimate and deviant signatures:

$$r(\tilde{\mathbf{b}}) = \frac{\sum_{\tilde{\mathbf{b}}_i \in nn(\tilde{\mathbf{b}}, \mathcal{B}_s)}^{k} d(\tilde{\mathbf{b}}, \tilde{\mathbf{b}}_i)}{\sum_{\tilde{\mathbf{b}}_i \in nn(\tilde{\mathbf{b}}, \mathcal{B}_n)}^{k} d(\tilde{\mathbf{b}}, \tilde{\mathbf{b}}_i)}, \tag{5.17}$$

where d is a distance measure (for example, Manhattan, Euclidean, or Mahalanobis distance) and nn is a set of signatures ordered by the distance to behavior signature $\tilde{\mathbf{b}}$:

$$nn(\tilde{\mathbf{b}}, \mathcal{B}) = \{\tilde{\mathbf{b}}_1, \tilde{\mathbf{b}}_2, \cdots, \tilde{\mathbf{b}}_{|\mathcal{B}|} | d(\tilde{\mathbf{b}}, \tilde{\mathbf{b}}_i) \leq d(\tilde{\mathbf{b}}, \tilde{\mathbf{b}}_{i+1})\}. \tag{5.18}$$

Practical applications are frequently faced with absence of one of the $\mathcal{B}_n$ or $\mathcal{B}_s$ databases. In this case, the detection is focused on one of the tasks: *suspicious* or *anomalous* behavior detection (Avrahami-Zilberbrand, 2009).

Definition 14. *Consider a set of behavior patterns $\mathcal{B}_s$ that encodes negative (suspicious) behavior. A behavior pattern $\mathbf{b} \notin \mathcal{B}_s$ is* suspicious *iff*

$$\exists \mathbf{b}_i \in \mathcal{B}_s : d(\mathbf{b}, \mathbf{b_i}) < \epsilon.$$

In other words, a behavior pattern is considered suspicious if it matches at least one suspicious behavior pattern. Since suspicious behavior is usually rare, this approach requires an expert that specifies all the suspicious patterns. Thus, such detection systems can only discover threats that are known *a priori*. Examples include models that encode possible attacks in intrusion detection systems (Biermann et al., 2001), trying to identify if an attack is under progress; and models with fall-dynamic templates in remote eldercare systems (Lee et al., 2004) that are compared to observation to infer whether the situation matches one or more templates.

The other approach uses a set of behavior patterns in an inverse fashion.

Definition 15. *Consider a set of behavior patterns $\mathcal{B}_n$ that encode positive (normal) behavior. A behavior pattern $\mathbf{b} \notin \mathcal{B}_n$ is* anomalous *iff*

$$\forall \mathbf{b}_i \in \mathcal{B}_n : d(\mathbf{b}, \mathbf{b_i}) > \epsilon'.$$

The set of behavior patterns is limited to covering only positive, expected behavior patterns. If a behavior pattern cannot be matched against those patterns, it is announced as anomalous. An advantage of this approach is that it requires only examples of positive behavior, which are usually abundant and attainable. This approach can thus detect unforeseen threats and contingencies. However, one potential issue is false positives caused by an incomplete legitimate behavior set; that is, a legitimate behavior pattern is marked as anomalous if it was not previously included in the set $\mathcal{B}_n$.

A notable approach is Local Outlier Factor (LOF) (Breunig et al., 2000), an outlier-detection algorithm based on computing the local neighborhoods densities. The main idea of the LOF algorithm is to assign to each signature a degree of being an outlier. This degree is the LOF of a vector. Vectors with a high LOF have local densities smaller than their neighborhood and typically represent stronger outliers, unlike vectors belonging to uniform clusters that usually tend to have lower LOF values. Due to the local approach, LOF is able to identify outliers in a data set that would not be outliers in another area of the same data set. For example, a point at a small distance to a very dense cluster is an outlier, while a point within a sparse cluster might exhibit similar distances to its neighbors.

The more behavior signature features, the more likely each signature is unique, and the less likely that any fixed size historical set represents a substantial mass of all signatures; that is, any density estimator becomes less reliable.

5.3.3 Machine Learning Approaches

Detectors employ statistical models generated from historical signatures automatically or by an expert. They specify relationships between the values of features and target value. For example, a rule might state if the landmark is *kitchen* and activity is *lying*, then signature is *suspicious.*

If no historical signature is available, an expert specifies a set of rules covering either desired (legitimate) or unwanted (deviant) behavior. Detection is performed analogously as in the previous section.

Data-driven rule generation is defined as a classification task, which constructs a model able to exploit statistical differences in a database of historical behavior signatures. Notable approaches are decision trees, SVMs, neural networks, etc.

5.4 Combining Multiple Detectors

As indicated in several studies, it is often advantageous to combine decisions of several experts to reach the final conclusion (Woods et al., 1996; Gams, 2001; Vilalta and Drissi, 2002). A single detector might be weak in certain aspect; that is, unable to reliably detect or evaluate specific behaviors. For example, consider the ambient assisted living domain. An anomalous event, such as fall, must be detected within seconds, while gradual degradation of gait resulting in limping might require months to be manifested. A detector specialized for anomalous events on the *seconds* scale would be unable to detect such degradation; analogously, a detector operating on *months* scale would simply filter out events such as falls. The combination of both would address both behaviors to provide anomalous events detection.

The main idea is to construct several local views; that is, detectors trained on a subset of complete feature space such as modalities differences, time scales, contextual information, and detection method approaches, into the final evaluation.

There are several possible strategies for performing the final evaluation: first, all detectors are considered equally important/reliable; second, accept the decision delivered by the most reliable expert ignoring majority consensus; third, weight detectors by their importance/past perfomance/reliability. A promising direction is to utilize expert knowledge to encode the last approach in a form of a Bayesian network as a set of random variables and their conditional dependencies via a directed, acyclic graph. For example, a Bayesian network could represent the probabilistic relationships between the behavior in question and detectors. Given detectors' evaluations, the network can be used to compute the probabilities that the behavior deviates.

5.5 Summary and Discussion

This chapter helped to understand the anomalous and suspicious behavior detection problem, and why it is difficult to solve. The chapter gave the first clear problem definition and established a theoretical framework for anomalous and suspicious behavior detection from agent traces. We showed how to optimally perform detection, discussed why detection error is often inevitable, and proved the lower error bound. We further provided several heuristic approaches that either estimated distributions required to perform detection or directly rank the behavior signatures using machine-learning approaches.

The main assumption of this chapter was that the decision must is based on a single agent observation. In practice, however, there are often many observations available, which allows the observer to make a decision based on multiple observations. The next chapter extends the theoretical framework to address multiple observations, discusses emerged issues, and proposes an approach to evaluate multiple observations.

6 Accumulating Behavior Evaluations Over Time

Anomalous and suspicious behavior detection becomes more challenging when agents are observed over a longer period of time. In many domains, no single event is sufficient to determine deviant behavior. Instead, multiple evaluations must be combined. This contrasts with previous chapters, which focused on the detection of a single, clearly suspicious event.

This chapter proposes a two-step detection system, which first detects trigger events in behavior trace, and then combines the evidence to provide a degree of suspicion. The chapter specifies conditions that any reasonable detector should satisfy, analyzes three detectors, and proposes a novel detector that generalizes a utility-based plan recognition with arbitrary utility functions.

6.1 Problem Statement

We target a large class of applications where no single event is sufficient to make a decision about whether or not behavior is suspicious. Instead, we face a sparse set of *trigger events* that identifies interesting parts characterizing the behavior trace.

Definition 16. Trigger event e_t *is an evaluation of a subsequence in behavior trace* $\mathbf{b}$*. It is described by probabilities that the corresponding subsequence is suspicious* $s(e)$ *and normal* $n(e)$*.*

Instead of constructing a behavior pattern from the complete behavior trace, only interesting subsequences are extracted and denoted as trigger events. Each event is then further analyzed as a behavior pattern and evaluated as described in the previous chapter. Multiple trigger events are combined into an event trace.

Definition 17. Event trace $\mathbf{e}^{(k)}$ *is a totally-ordered sequence of* k *trigger events* $\mathbf{e}^{(k)} = \{e_1, e_2, ..., e_k\}$*.*

Examples include a potentially suspicious passenger who appears to turn away in the presence of security personnel, but not blatantly so; hence, no single event is enough to raise suspicion. The main question we address is how to combine multiple events to decide whether an event trace corresponds to the normal or a suspicious agent behavior.

We address the suspicious behavior detection problem in two steps, as shown in Figure 6.1. The first step analyzes an action trace and the surrounding environment to detect trigger events that characterize its interesting parts. The event trace is then evaluated in second step. If the evaluation exceeds a threshold value or is large relative to other evaluations, the event is considered suspicious.

The key contributions of this chapter are in the second step, which is defined as a decision problem: is the behavior of an agent suspicious given a sequence of trigger events? First, we formally describe the detection problem and specify the conditions that any reasonable detector should satisfy. Second, we analyze three detectors, namely the naïve Bayes detector, the hidden Markov models and the utility-based plan recognition (UPR). These detectors, however, either simplify the problem or evaluate the events linearly. Finally, we present

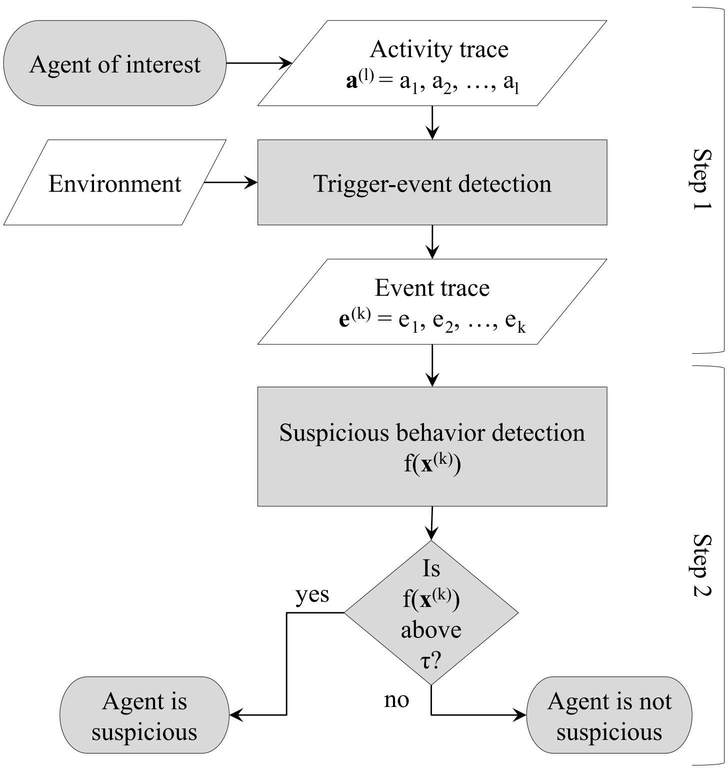


Figure 6.1: Two-step detection of suspicious behavior: (i) detection of trigger events; and (ii) detection of suspicious behavior.

a novel detector that generalizes UPR as Function-UPR (F-UPR): we define utilities as a set of functions over state transitions and observations, and introduce an observation utility function that is especially suitable for suspicious behavior detection, since it is able to evaluate events non-linearly.

6.2 Detection Scheme

This section formally analyzes how to evaluate a sequence of trigger events. Our methods are general, but we will make use of the airport domain (Chapter 9) to provide examples, where the goal is to detect a suspicious passenger from a surveillance camera network. Even though we will be focused on suspicious behavior detection, the same conclusion can be drawn for anomalous behavior detection.

We leverage the Bayesian intrusion-detection framework (Helman and Liepins, 1993) to define the problem. At each time step t, we observe an event e_t, generated by a hidden stochastic process H. Now suppose that H is a mixture of two auxiliary stochastic processes, namely the normal process N and the suspicious process S. The random variable $y_t = 0$ if e_t is generated by N and $y_t = 1$ if e_t is generated by S. Since a suspicious passenger always emits a suspicious event (and a normal person a normal event), y for a specific agent does not change over time. In reality, there can be many subprocesses contributing to both N and S; that is, many normal persons with different behavior patterns. However, here we assume only a single N and a single S that capture all the variability.

To this point, we assumed that an observer is able to perfectly observe whether an event is generated by S or N. In practice, however, it may appear that a normal person emits suspicious events (or vice-versa). An observer might be limited for various reasons, such as an inability to detect characterizing features and noisy trigger-event detectors. Therefore, we relax this assumption as follows. An event e_t is observed as generated by N with the probability

$$n(e_t) = \Pr\{H(t) = e_t | y_t = 0\} \tag{6.1}$$

and as generated by S with the probability

$$s(e_t) = \Pr\{H(t) = e_t | y_t = 1\} = 1 - n(e_t). \tag{6.2}$$

The mixture distribution of an event e_t and a prior probability λ is

$$\Pr\{H(t) = e_t\} = \lambda s(e_t) + (1 - \lambda) n(e_t). \tag{6.3}$$

The objective of suspicious behavior detection is to identify those traces $\mathbf{e}^{(k)} = (e_1, e_2, ..., e_k)$ that are likely to be suspicious activities; that is, traces $\mathbf{x}$ for which

$$\Pr\{y = 1 | H(t) = e_t, t = 1, ..., k\} > \tau, \tag{6.4}$$

is above some threshold τ or is large relative to the probability for other traces.

The reason this problem is difficult is the non-linear effect. Consider the following example: suppose we observe a person do a U-turn in front of a police officer, so such that the likelihood that this was a suspicious person becomes high. Later, we see the same person doing a half-turn in front of a police officer. This trigger event, if seen on its own, would not contribute much to the overall suspicion. However, following the initial observed turn, this new turn is a much stronger evidence to be attributed to the overall suspicion, because we bias the new event with our previous observation.

Theoretically, it might be possible to optimally detect suspicious behavior using Equation (6.4). Unfortunately, this is usually not the case in practice. To see this, let us assume a prior probability $\lambda = \Pr\{y_t = 1, t = 1, ..., k\}$. In most cases, λ is close to 0, since in real-world applications suspicious activities are rare. Let the stochastic processes N, S and H denote $n(\mathbf{e}^{(k)}) = \Pr\{H(t) = e_t, t = 1, ..., k | y = 0\}$, $s(\mathbf{e}^{(k)}) = \Pr\{H(t) = e_t, t = 1, ..., k | y = 1\}$, and $h(\mathbf{e}^{(k)}) = \Pr\{H(t) = e_t, t = 1, ..., k\}$, respectively. Using Bayes theorem we can derive from Equation (6.4)

$$\Pr\{y = 1 | H(t) = e_t, t = 1, ..., k\} = \frac{\lambda \cdot s(\mathbf{e}^{(k)})}{h(\mathbf{e}^{(k)})} = \tag{6.5}$$

$$= \frac{\lambda \cdot \prod_{t=1}^{k} s(e_t | e_{i, i=t-1,...,1}\,)}{\lambda \prod_{t=1}^{k} s(e_t | e_{i, i=t-1,...,1}\,) + (1 - \lambda) \prod_{t=1}^{k} n(e_t | e_{i, i=t-1,...,1}\,)}.$$

To this point, we implicitly assumed that the distributions λ, n, and s are reliably estimatable. The degree to which this assumption is valid depends on our detection capability. Suppose we have a sufficiently large dataset $\mathcal{D}_l$ of labeled event traces: we can estimate the prior probability λ using the relative frequency, presenting the number of traces generated by a suspicious agent divided by the total number of traces (since traces can be of different lengths, the quotient is normalized by the traces' length). Note that in order to compute $\Pr\{H(t) = e_t, t = 1, ..., k | y = 1\}$ we have to evaluate

$$s(e_1) \cdot s(e_2 | e_1) \cdot ... \cdot s(e_k | e_{k-1}, ..., e_1). \tag{6.6}$$

While some first terms, that is, $s(e_t), s(e_t | e_{t-1})$, can still be estimated, latter term estimation, including increasingly more history, becomes less and less reliable. In real-world applications, we have no direct knowledge of the conditional probabilities values; that is, we are unable to specify the probability of an event given all the possible combinations of history. For this reason, we must approximate the Bayes optimality in general. In particular, we will be concerned with estimating $\Pr\{y = 1 | H(t) = e_t, t = 1, ..., k\}$ using approximate approaches.

Given an event trace, some events may appear suspicious and some not. Hence, detection systems must have a scoring function that combines the evidence. Function output is interpreted as the degree of suspicion attributed to the event trace. Although any two scoring functions need not be exactly the same, we can specify the conditions that any reasonable scoring function must satisfy. The class defined below appears to be both natural and general.

The detection system can employ a *scoring function* f that interprets events to produce a score characterizing the overall suspicion of the trace. Given a threshold value τ and an event trace $\mathbf{e}^{(k)}$, we can classify $\mathbf{e}^{(k)}$ as suspicious if $f(\mathbf{e}^{(k)}) \geq \tau$.

Definition 18. *A scoring function f over a trace of events $\mathbf{e}^{(k)}$ is a function*

$$f : \bigcup_{k=1}^{K} \mathbf{e}^{(k)} \to \mathbb{R}.$$

The function f assigns a real value to any trace $\mathbf{e}^{(k)}$ of length $k = 1, ..., K$.
Let $\Delta(e_t)$ decide whether a single event e_t is suspicious or not given a threshold τ':

$$\Delta(e_t) = \begin{cases} 1; & \text{if } s'(e_t) \geq \tau' \\ 0; & \text{else} \end{cases}, \tag{6.7}$$

$$s'(e_t) = \frac{\lambda \cdot s(e_t)}{\lambda \cdot s(e_t) + (1 - \lambda) \cdot n(e_t)}. \tag{6.8}$$

Definition 19. *A class of* well-behaved *functions consist of scoring functions s.t.* $\forall \mathbf{e}^{(k)}, e_{k+1}$:

$$\begin{aligned} f(\mathbf{e}^{(k)}, e_{k+1}) \geq f(\mathbf{e}^{(k)}) \qquad & \textit{if } \Delta(e_{k+1}) = 1, \\ f(\mathbf{e}^{(k)}, e_{k+1}) \leq f(\mathbf{e}^{(k)}) \qquad & \textit{if } \Delta(e_{k+1}) = 0. \end{aligned}$$

The conditions imply that the scoring function f's evaluation increases when a new suspicious event is added to the trace, and decreases when a normal event is added to the trace. The well-behaved scoring functions are motivated by the key observation that a suspicious event e_{k+1} (that is, $\Delta(e_{k+1}) = 1$) is more likely to be generated by a suspicious process S than a normal process N, regardless of the history $\mathbf{e}^{(k)}$, that is,

$$\begin{aligned} s(e_{k+1} | \mathbf{e}^{(k)}) \geq n(e_{k+1} | \mathbf{e}^{(k)}) \qquad & \text{if } \Delta(e_{k+1}) = 1 \text{ and} \\ s(e_{k+1} | \mathbf{e}^{(k)}) \leq n(e_{k+1} | \mathbf{e}^{(k)}) \qquad & \text{if } \Delta(e_{k+1}) = 0. \end{aligned}$$

6.3 Detectors

This section analyzes approaches that determine whether an event trace is suspicious. First, we discuss the naïve Bayes detector that relaxes the initial assumptions. Next, we discuss an approach that directly tackles estimating the likelihood that a trace was generated by a suspicious process using HMMs. Finally, we analyze an approach based on plan recognition and present two extensions: (i) we define utilities as a potential function; and (ii) we present an observation utility function able to address non-linear accumulation.

6.3.1 Naïve Bayes Detector

A naive approach assumes that events are independent, which means that the current event depends only on the current time step t and not on the time steps prior to t. The evaluation of Equation (6.5) is simplified using the naive assumption:

$$\Pr\{y = 1|H(t) = e_t, t = 1, ..., k\} = \frac{\lambda \cdot \prod_{t=1}^{k} \hat{s}(e_t)}{\lambda \cdot \prod_{i=1}^{k} \hat{s}(e_t) + (1-\lambda) \cdot \prod_{i=1}^{k} \hat{n}(e_t)}. \tag{6.9}$$

We have to evaluate the probability $\Pr\{H(t) = e_t|y_t\}$ that an event is generated by both a normal process $\hat{n}(e_t)$ and a suspicious process $\hat{s}(e_t)$, which is tractable in terms of evaluation. The approaches for estimating $\hat{n}$ and $\hat{s}$ may include a frequentist estimator, hidden Markov models, k-nearest neighbors, neural networks, etc. We showed an approach using CHMM in Section 3.6. An evaluation of the event trace is also well behaved when $\tau' = \lambda$.

In practice, the model may be oversimplified by the assumptions; however, we will use it as a baseline in our experiments.

6.3.2 Hidden Markov Models

A conditional probabilities estimation including the history can be encoded with HMMs (Rabiner, 1989), as described in Section 3.4.2. Now suppose we create an HMM to estimate $\Pr\{H(t) = e_t|y = 1, t = 1, ..., k\}$; more precisely, it models the probability that an event trace is generated by a suspicious agent. The hidden states of the process Q may be referred to as internal states presenting the suspicious agent's intentions. For the sake of clarity, let us assume only two hidden states: a normal intention and a suspicious intention, emitting normal and suspicious events, respectively. The transitions between the hidden states can be explained as probabilities that the agent will either follow or change its current intention. Informally, this intention switching may be interpreted as follows: from an observer's perspective, sometimes suggesting that the observed agent is switching intentions appears to provide a better explanation of the behaviors.

We construct two HMM models: a normal model $\bar{N}$ and a suspicious model $\bar{S}$. We split all the labeled event traces $\mathbf{e} \in D_l$ to traces generated by normal and suspicious agents, and use them to learn the parameters of the models $\bar{N}$ and $\bar{S}$, respectively. The parameters can be locally optimized using an iterative procedure such as Baum-Welch method (Rabiner, 1989). Given a new event trace $\mathbf{e}^{(k)} = (e_1, e_2, ..., e_k)$, we compute the probability that the trace was generated by each model $\Pr\{\mathbf{e}^{(x)}|\bar{N}\}$ and $\Pr\{\mathbf{e}^{(x)}|\bar{S}\}$ using a forward-backward procedure (Rabiner, 1989). Given the prior probability $\bar{\lambda} = \Pr\{y_t = 1, t = 1, ..., k\}$, we estimate the trace $\mathbf{e}^{(k)}$ was generated by the suspicious process S:

$$\Pr\{y = 1|H(t) = e_t, t = 1, ..., k\} = \frac{\bar{\lambda} \cdot \Pr\{\mathbf{e}^{(k)}|\bar{S}\}}{\bar{\lambda} \cdot \Pr\{\mathbf{e}^{(k)}|\bar{S}\} + (1-\bar{\lambda}) \cdot \Pr\{\mathbf{e}^{(k)}|\bar{N}\}}. \tag{6.10}$$

Although the information about previous behavior is now partially encoded in the transition probabilities (that is, given that the agent's intention at time step t is suspicious, it is more likely that the intention at $t + 1$ will be suspicious as well), the model still uses the Markov assumption; that is, the next agent's intention depends only on its current intention. It is possible to introduce more complex HMM structures with long-term dependencies, but learning and inference in such models becomes computationally intractable (Koller and Friedman, 2009).

6.3.3 Utility-Based Plan Recognition

We exploit UPR, a *Utility-based Plan Recognition*, briefly described below. The reader is referred to Avrahami-Zilberbrand and Kaminka (2007) for details. UPR consists of a plan library, which encodes observed agent behaviors in a form of directed graph, and a matching algorithm. It follows the footsteps of the hierarchical HMM in representing probabilistic information in a plan library. A plan step can be atomic, or non-atomic; that is, broken down into atomic sub-steps, each a plan step in itself. Plan steps are linked via sequential edges, describing the execution order of a given plan and its sub-steps. UPR introduces three types of utilities on the edges: (i) the sequential utility from the current step to the next; (ii) the interruption utility from the current step to the end of the plan; and (iii) the decomposition utility from the current step at current level to its first substep at the sub-level. A corresponding probability is maintained for each type of utility. The observation sequence o is matched against the library using a *Symbolic Plan Recognizer* (Avrahami-Zilberbrand, 2009), which filters hypotheses that are consistent with o. Finally, the hypotheses are ranked by their expected utility.

We use a heuristic version of UPR as follows: let $\hat{s}(e_t) = 1 - \hat{n}(e_t)$ be the probability that the trigger event e_t was generated by a suspicious person. Let $c_s > 0$ be the cost of the damage caused by a suspicious person if we do not stop him, and, similarly, let $d_n = 0$ be the cost of the damage caused by a normal person. The expected cost of letting this person go (marking him as normal) is $c_{go} = c_s\hat{s}(e_t) + d_n\hat{n}(e_t) = c_s\hat{s}(e_t)$. Now suppose $c_n > 0$ is the cost of arresting an innocent person and $d_s = 0$ is the cost of the damage caused by a suspicious person when arrested. The expected cost of stopping this person (marking him as suspicious) is $c_{stop} = c_n\hat{n}(e_t) + d_s\hat{s}(e_t) = c_n\hat{n}(e_t)$. If there was only one event, we would compare both hypotheses and choose the one with the lowest expected cost. Supposing in this case $c_n\hat{n}(e_t)$ is lower, we would call this person suspicious.

One possible approach, based on the above expected-cost calculation, would be to determine whether to categorize a trigger event as suspicious or normal, and then to accumulate the total number of suspicious events, and subtract the total number of normal events; unfortunately, this simple strategy performs poorly. Therefore, not only do we count whether an event is suspicious or normal, but we give it a weight proportional to the benefit or cost accrued. The function U_{UPR} then evaluates an event trace $\mathbf{e}^{(k)}$ of a person by accumulating the weighted benefit of stopping this person and subtracting the weighted cost of arresting a normal person:

$$U_{UPR}(\mathbf{e}^{(k)}) = \sum_{t=1}^{k} u(e_t), \tag{6.11}$$

$$u(e_t) = \begin{cases} c_s\hat{s}(e_t); & \text{if } c_n\hat{n}(e_t) \leq c_s\hat{s}(e_t) \\ -c_n\hat{n}(e_t); & \text{if } c_n\hat{n}(e_t) > c_s\hat{s}(e_t) \end{cases}. \tag{6.12}$$

If the accumulated cost exceeds a threshold value τ', the person (that is, trace $\mathbf{e}^{(k)}$) is marked as suspicious.

This remains a heuristic approach and further investigations could be a topic for future work; however, given that our next approach has significantly superior results, we chose to investigate that in more detail rather than providing more heuristics for the current approach.

6.4 Utilities as Potential Functions

Although the evaluation function U_{UPR} is well behaved, the utilities are constant and hence do not allow a dynamic adjustment for past agent behavior; for instance, the first time we note a suspicious event counts equally, with subsequent suspicious events made by the same agent. These utilities, however, are unable to express the empirical observation characteristics. Therefore, we extend the notion of utility and define the utility U as follows.

Definition 20. *The utility function U over a plan step q_a, a plan step q_b, and the entire observation sequence $\mathbf{e}^{(t)}$ until current time step t is a function*

$$U : \langle q_a, q_b, \mathbf{e}^{(t)} \rangle^n \to \mathbb{R}.$$

Utility function can be written as

$$U(q_a, q_b, \mathbf{e}^{(t)}) = \sum_{j=1}^{n} \lambda_j u_j(q_a, q_b, \mathbf{e}^{(t)}),$$

where each utility function u_j can be sequential, interruption, decomposition or any other utility, and λ_j are parameters to be defined. This allows us to introduce a set of auxiliary utility functions u_j describing not only the plan-step transitions but also the additional characteristics of the observation sequence. For example, the sequential utility from step q_i to q_{i+1} can be written as $u_t(q_i, q_{i+1}, \mathbf{e}^{(t)}) = c$, but in general, the constant c can be replaced with any function over q_i, q_{i+1} and $\mathbf{e}^{(t)}$.

Theorem 2. *U is a well-behaved function iff*

$$\forall u_j, j = 1...k : u_j \textit{ is a well behaved function.}$$

Proof. Consider two well behaved functions f and g, and two scalar constants λ_f and λ_g. Let $f' = \lambda_f f$. Since it is easy to see that multiplication with scalar constants preserves the well-behaved property, f' is also a well behaved function. Let function u denote $u = f' + g'$. Then, $u(\mathbf{e}^{(t)}, e_{t+1}) = f'(\mathbf{e}^{(t)}, e_{t+1}) + g'(\mathbf{e}^{(t)}, e_{t+1}) \geq u(\mathbf{e}^{(t)}) = f'(\mathbf{e}^{(t)}) + g'(\mathbf{e}^{(t)})$ if $\Delta(e_{t+1} = 1)$, since f and g are well behaved and therefore $f'(\mathbf{e}^{(t)}, e_{t+1})$ and $g'(\mathbf{e}^{(t)}), e_{t+1})$ are non-negative. Similarly, f' and g' are non-positive when $\Delta(e_{t+1}) = 0$. □

6.4.1 Exponential Observation Utility

In order to include agent past behavior in an evidence evaluation, the utility function must be defined over the observation sequence. We propose an observation utility function that assigns cost using the number of past normal and suspicious events. Consider the example from Section 6.1. Suppose we see a person do a full U-turn in front of a police officer and we give this event a cost of 1. Later, we see the same person doing a half-turn in front of a police officer. This event if seen on its own, would be given cost 0.5. However, following this initial turn, this new turn becomes a 1 instead of 0.5. So, a linear accumulation would have given us a cost of 1.5, whereas because we bias the new event to register higher on our scale, our cost is 2 instead of 1.5.

Let $\eta_s(\mathbf{e}^{(k)})$ define the number of suspicious events in an event trace $\mathbf{e}^{(k)}$:

$$\eta_s(\mathbf{e}^{(k)}) = \sum_{t=1}^{k} \Delta(e_t). \tag{6.13}$$

Similarly, let $\eta_n(\mathbf{e}^{(k)}) = k - \eta_s(\mathbf{e}^{(k)})$ represent the number of normal events. Suppose we observed a trace $\mathbf{e}^{(k)}$ of all the suspicious events; that is, $\forall t, t = 1, ..., k : \Delta(e_t) = 1$. Intuitively, the likelihood that an event e_t was indeed generated by a suspicious process increases exponentially according to the number of suspicious events in the past. On the other hand, if the events in $\mathbf{e}$ were normal; that is, $\forall t, t = 1, ..., k : \Delta(e_t) = 0$, the likelihood decreases exponentially as the number of normal events increases. We define an observation utility function u_o over the current event e_t and trace $\mathbf{e}^{(t-1)}$ recursively as follows:

$$\begin{aligned} u_o(e_t, \mathbf{e}^{(t-1)}) &= \psi(\mathbf{e}^{(t)}) \cdot (u_o(\mathbf{e}^{(t-1)}) + \omega(\mathbf{e}^{(t)})), && (6.14) \\ u_o(\mathbf{e}^{(0)}) &= 0, \\ \omega(\mathbf{e}^{(t)}) &= \alpha \cdot \eta_s(\mathbf{e}^{(t)})^{s(e_t)/\beta}, && (6.15) \\ \psi(\mathbf{e}^{(t)}) &= \gamma \cdot \rho^{-\eta_n^*(\mathbf{e}^{(t)})/\eta_s(\mathbf{e}^{(t)})}. && (6.16) \end{aligned}$$

The term $\omega(\mathbf{e}^{(t)})$ uses an exponential function to assign a cost to the likelihood $s(e_t)$ that an event is suspicious. The parameter $\alpha > 0$ is the initial cost, η_s corresponds to the growth factor, and the parameter $0 < \beta < 1$ is the likelihood of the cost increasing by the growth factor. The parameters α and β are estimated from the data. In the case of observing two full U-turns, the second U-turn attributes higher cost to the overall suspicion, since the exponent base is increased due to the first U-turn.

Additionally, the term $\psi(\mathbf{e}^{(t)})$ employs an exponential time decay function that discounts the accumulated cost at time t according to the number of consecutive normal events η_n^*. The modified η_n^* represents *the time elapsed* since the last event $\Delta(e_i) = 1$; that is, the number of normal events since the last suspicious event. The higher the number of consecutive normal events, the faster the cost decay. The parameter $0 < \gamma \leq 1$ is the initial decay, the parameter $0 < \rho < 1$ is the decay factor, and η_s is used to specify the number of events required for the decay to decrease by the decay factor. The parameters γ and ρ are also estimated from the data. Suppose we observe two agents, one already having made two U-turns and the other having made a single U-turn. Suppose, then, we observe both agents do a clearly normal event; the overall suspicion of the first agent is less than the overall suspicion of the second agent. Hence, the higher the number of suspicious events, the slower the suspicion decay.

The function u_o is a well-behaved function by definition. Equation (6.14) can be rewritten, which gives us the utility function U_{F-UPR}:

$$\begin{aligned} U_{F-UPR}(\mathbf{e}^{(k)}) &= \sum_{t=1}^{k}\sum_{j=1}^{n} \lambda_j f_j(\mathbf{e}^{(t)}, q(t-i), q(t)) \\ &= \sum_{t=1}^{k}(\omega(\mathbf{e}^{(t)}) \prod_{i=t}^{k} \psi(\mathbf{e}^{(i)})). && (6.17) \end{aligned}$$

6.5 Summary and Discussion

This chapter extended the theoretical framework established in the previous chapter. It showed how to perform detection when an agent is observed over longer periods of time and no significant event is sufficient to reach decision. We first specified conditions any reasonable detector should satisfy and analyzed several detectors. We further proposed a novel approach denoted as F-UPR detector that generalizes UPR (Avrahami-Zilberbrand and Kaminka, 2007) with arbitrary utility functions. This allows to assign utility to repeated plan steps according to agent past behavior, which was, in turn, used to introduce an exponential observation utility function that assigns cost using the number of agent's past normal and suspicious events.

7 A Unified Detection Framework

In this chapter, a unified framework for anomalous and suspicious behavior detection is presented. The goal is to design a framework rich enough to address various domains and different configurations. Therefore, we summarize its components and explain their relationships. The next part of the thesis then shows how to instantiate the detection framework in various real-life domains.

7.1 Framework Components

All the components presented in the previous chapters form the unified framework for anomalous and suspicious behavior detection. This section reviews the components, their hierarchical structure, and explains the processing steps of the unified framework.

7.1.1 Framework Levels

Let us revisit the framework level hierarchy shown in Figure 7.1 that systematically processes agent spatio-temporal data to perform deviant behavior detection. The information flow is bottom-up through three main levels shown on the left-hand side: measurements, activity assessment, and behavior assessment. Each of the main levels contains several sub-levels (right-hand side): 13 in total.

The measurement level provides sensor data, which are collected at each time step t in the next sub-level as an *observation vector* $\mathbf{x}_t$. In addition to the sensor data, the observation vector may also contain contextual information and environmental variables such as time, date, weather, etc. Subsequent observation vectors are bundled in an observation sequence $\mathbf{X}$.

The next main level is activity assessment. The first four sub-levels basically correspond to ARPipe (activity recognition pipeline, Chapter 3), which first suppresses the sensor noise, followed by the construction of activity recognition feature vectors and atomic activity recognition itself. Finally, the spurious transitions among atomic activities that cannot occur in reality are smoothed. *Activity sequence* containing an atomic action at each time step $\mathbf{a}^{(T)} = \{a_t | 1 \leq t \leq T\}$ is passed to the next sub-level, which can recognize complex behaviors or interactions among agents.

The top level performs behavior assessment in order to detect deviance. The first sub-level constructs a *behavior trace* that consists of $\langle landmark, activity \rangle$ tuples, which are encoded as a *behavior pattern* in the next sub-level. The next sub-level then introduces a set of detectors based on different features, views and modalities to provide different evaluations of the behavior pattern. The evaluations are then combined at each time step in the next sub-level as well as over time at the level above. Finally, the top sub-level outputs the final agent behavior evaluation that comprehends all the behavior to the current time step.

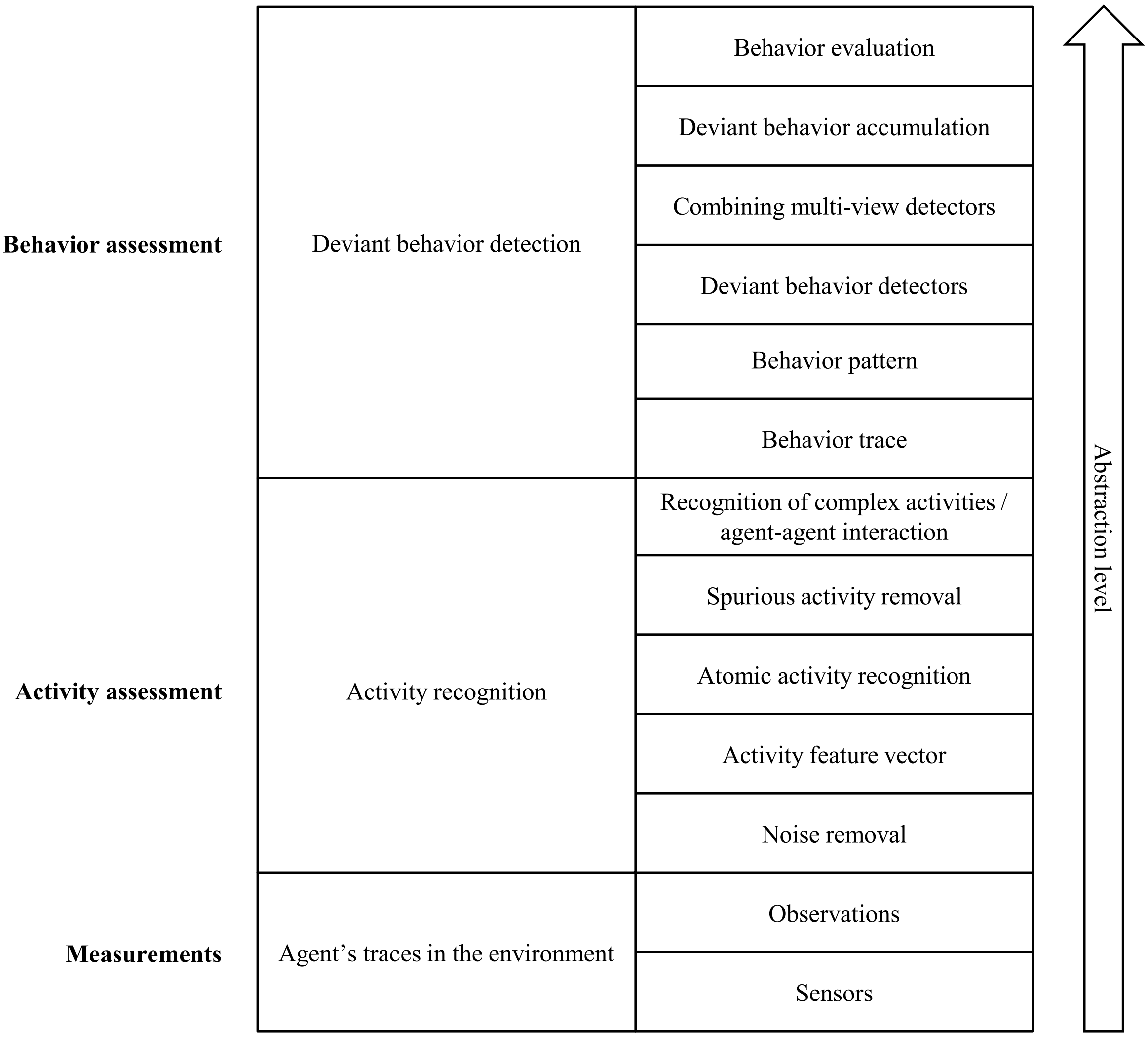

Behavior assessment
Deviant behavior detection
Behavior evaluation
Deviant behavior accumulation
Combining multi-view detectors
Deviant behavior detectors
Behavior pattern
Behavior trace
Activity assessment
Activity recognition
Recognition of complex activities / agent-agent interaction
Spurious activity removal
Atomic activity recognition
Activity feature vector
Noise removal
Measurements
Agent's traces in the environment
Observations
Sensors
Abstraction level

Figure 7.1: Hierarchy of abstraction levels and processes.

7.1.2 Processing Steps

A detailed unified framework flowchart is outlined in Figure 7.2. The start and the end of the process are shown in rounded rectangles, processes are represented as rectangles, and input/output data are represented as a parallelogram.

The process starts with an observation sequence describing agent movements in the environment. The trace is first processed by the activity recognition pipeline as described above and outputs an activity sequence. The activity sequence then enters deviant behavior detection level marked with dashed-squared box. The activity sequence is first augmented to behavior trace, which then enters into a variety of view transformation processes. Each view-transformation process applies its specific viewpoint using either different features, modalities, or time aggregation periods to construct corresponding behavior signatures. Each behavior signature is then evaluated with a deviant behavior detector separately. The next step combines all the evaluations that are further processed in a suspicious behavior accumulation module over time, which finally outputs the input trace deviation.

General implementation principles and the theoretical background of particular flowchart processes were discussed in the previous chapters, while the concreted domain implementation is demonstrated in Part II of this thesis. Note that, in some domains, not all the processes are required; a domain problem may simplify a certain process. For example, if only a single behavior view is required, the multi-view detector combination is simplified accordingly.

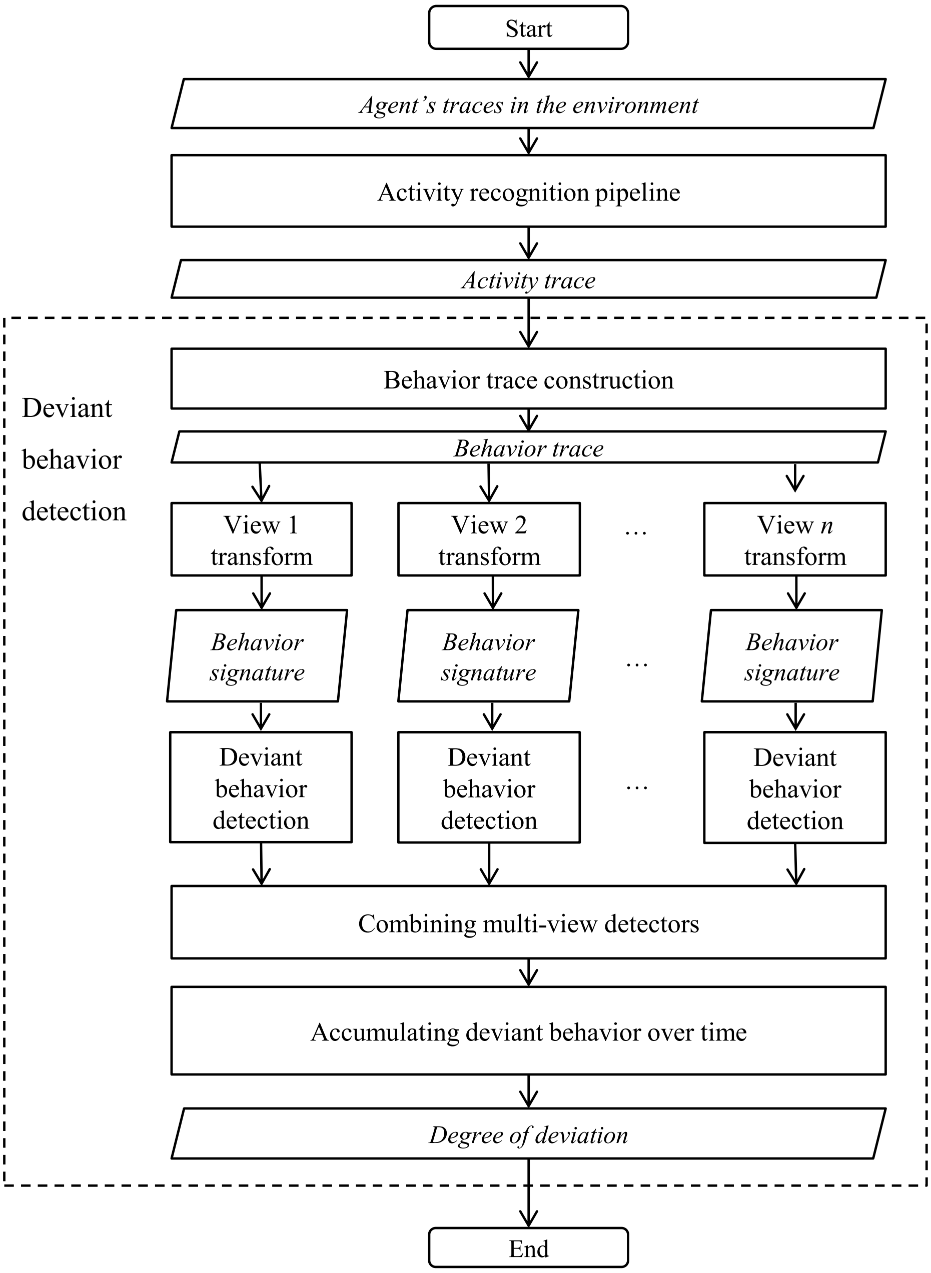

Start
Agent's traces in the environment
Activity recognition pipeline
Activity trace
Deviant behavior detection
Behavior trace construction
Behavior trace
View 1 transform
View 2 transform
...
View n transform
Behavior signature
Behavior signature
...
Behavior signature
Deviant behavior detection
Deviant behavior detection
...
Deviant behavior detection
Combining multi-view detectors
Accumulating deviant behavior over time
Degree of deviation
End

Figure 7.2: Processing flowchart of the unified framework.

7.2 Framework Instantiation

As will be shown in the next part of the thesis, the framework can be instantiated for various domains and problems. This section covers high-level framework instantiation, emphasizing the learning and detection phases.

7.2.1 Learning and Detection

Unified framework instantiation includes selecting and designing some domain-specific components as well as implementing general components described in the previous chapters. Since this will be covered in Part II of this thesis, these components will be abstracted and the focus will be on the learning and detection phase.

Figure 7.3 depicts a high-level block diagram of the learning (left-hand side) and the detection (right-hand side) phase within the unified framework. The goal of the learning phase is to instantiate all the components, which includes building classifiers and detectors, discovering patterns, and fine-tuning the parameters of the models. After that, the framework is deployed in the detection phase, which is dedicated to evaluating new traces at the input.

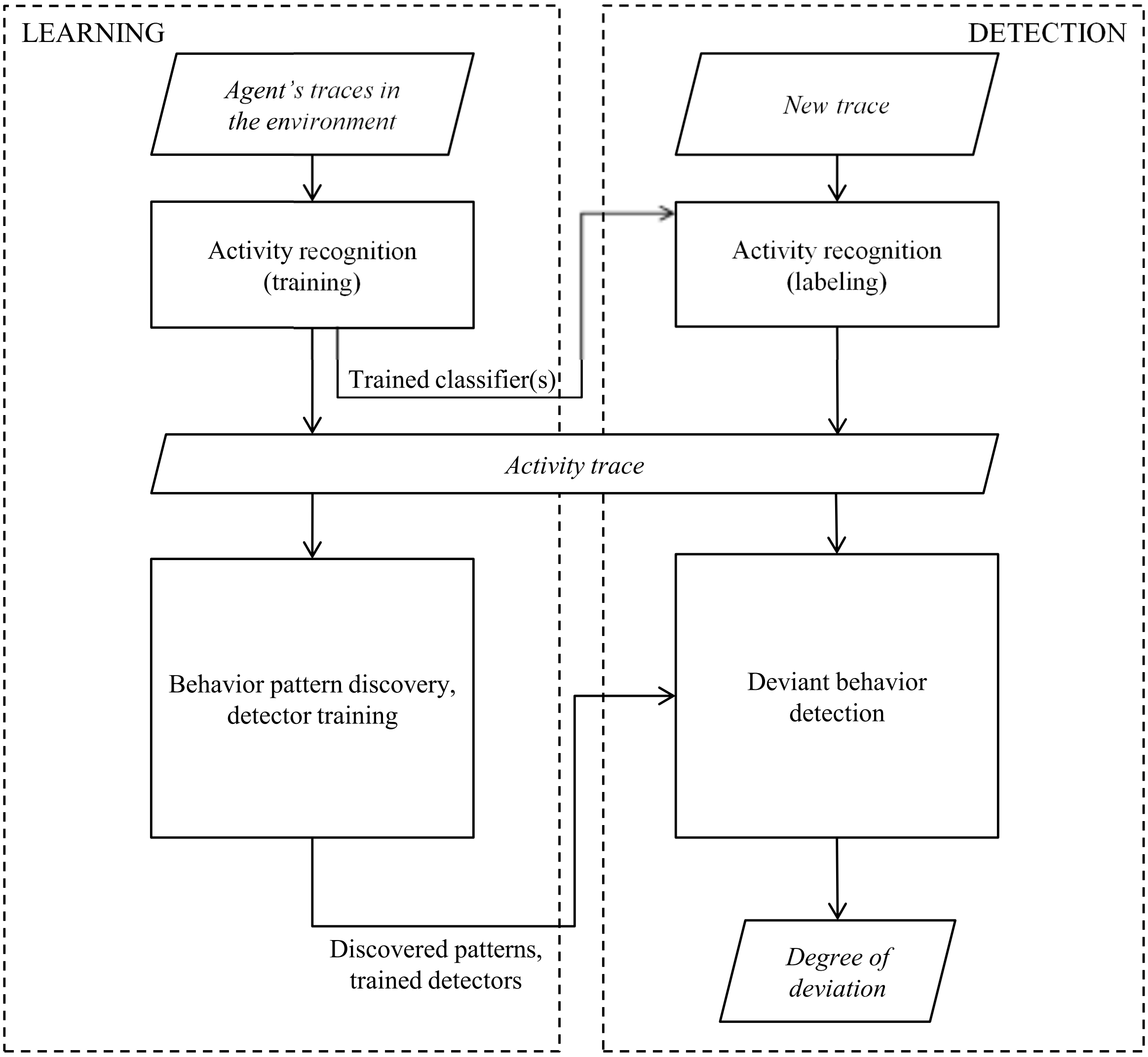

LEARNING
DETECTION
Agent's traces in the environment
New trace
Activity recognition (training)
Activity recognition (labeling)
Trained classifier(s)
Activity trace
Behavior pattern discovery, detector training
Deviant behavior detection
Discovered patterns, trained detectors
Degree of deviation

Figure 7.3: Block diagram for learning and detection phases.

The learning phase includes two components that require training. The first component is activity recognition, which, unless the activities are already provided from the environment, requires classifier training. This includes recording labeled training data to construct a training dataset, feature extraction, and building a classification model. Once the model is trained, it can be used in the detection phase.

The second component is dedicated to behavior patterns construction and detection model training. According to our detection goals, that is, either anomalous or suspicious behavior detection, the aim of this phase is to construct a dataset of positive or negative behavior patterns, respectively, using either automatic/semi-automatic approaches, such as clustering, or domain expert knowledge that encodes behavior patterns. As shown in Figure 7.2, this can be applied for a variety of viewpoints. The next step then includes detector training and initial parameter tuning, which also requires a training dataset. Note that the second component already requires that the activity recognition component is trained.

7.3 Summary

In summary, we proposed a unified framework for detection of anomalous and suspicious behavior that can be observed from complex, spatio-temporal sequential data generated by an agent moving in a physical environment. The framework incorporates the components introduced in the previous chapters to address the main challenges. The second part of the thesis shows how the framework is instantiated in three empirical studies.

Part II

Empirical Studies

8 Ambient Assisted Living Domain

Analyzing daily-living behavior is an important approach to assessing the wellbeing of an elderly person living at home alone. This chapter presents an approach to monitoring an individual in the home environment by an ambient-intelligence system to detect daily living pattern anomalies. It utilizes the proposed unified framework to recognize activities, extract spatio-activity behavior signatures, and apply an outlier-detection method to classify the individual's daily patterns, regardless of the cause of the problem, be it physical or mental. Experiments indicate that the proposed solution successfully discriminates between healthy person behavior patterns and those of a person with health problems.

8.1 Introduction and Background

Recent years have seen increased interest in the deployment of systems for ambient assisted living (AAL) (Augusto et al., 2012), including remote eldercare (Kaluža et al., 2010b), smart homes (Cook, 2009), surveillance (Dore et al., 2011), etc. Whereas some of these systems can be tele-operated, the AAL community strives to design systems that monitor a person autonomously and act in the case of an emergency, warning or suggestion, such as fall detection (Bourke and Lyons, 2008; Luštrek et al., 2009). Our study targets persons in the home environment; that is, a male or female senior citizen, who does not need intensive care or assistance in day-to-day living, but accepts an ambient-intelligence (AmI) system to improve their health, safety, and well-being. The main issue is anomaly detection in the monitored person's daily behavior.

A predominant approach consists of three components: a sensor system, an activity-recognition model, and a daily behavior analysis (Choudhury et al., 2006). There are several challenges to constructing such a system. First, the person must be monitored with sensors that are not obtrusive, invasive or privacy-violating, yet are precise enough to address the second challenge, which is an accurate activity recognition model. An underlying recognition model needs to detect a wide variety of activities performed differently under different environmental conditions and across many individuals. Third, we have no knowledge about the exact plans and schedules a person may follow during the day. In addition, the system should adapt to each specific person while deployed at the person's home.

In remote eldercare, the AAL systems use a wide variety of sensors, such as vision systems (Cardinaux et al., 2011), inertial sensors (Bourke and Lyons, 2008) and embedded sensors (Lymberopoulos et al., 2008; Monekosso and Remagnino, 2010). While some sensors might violate privacy issues, for example, a camera, others do not provide additional location context, for example, inertial sensors, or rich information required for accurate activity and posture recognition, for example, embedded sensors. Daily behavior analysis may focus on recognizing or describing exact schedules and assumes that the person will follow them. Another approach relies on either observers, that is, a nurse who periodically observes an elderly person, or on self-reporting, that is, having people complete an activity report at the end of the day. Both ways of reporting have limited accuracy and usefulness due to the aggregation in time, forgetfulness, and misreporting (intentional or unintentional).

In contrast to related work discussed in Section 2.2.2; that is, fuzzy-association analysis (Lee et al., 2004), Apriori algorithm combined with Markov chain (Lymberopoulos et al., 2008), and HMMs (Monekosso and Remagnino, 2010), this chapter uses a localization system (in other publications, accelerometers are more often used) with wireless body-worn tags (described in Section 8.3), while low-level activity recognition is performed with a SVM classifier. These two modules were developed within the Confidence system (Kaluža et al., 2010b). This chapter focuses on the third component; that is, daily patterns analysis that detects behavior changes indicating an early discovery of a potential health problem, such as a person visiting a toilet unusually often. In contrast to related work, which mainly dealt with a description of high-level activities, our method focuses on activity dynamics; by contrast to Markov models, it explores the relations between spatial information and activities. The method is general in the sense that it detects unusual behavior regardless of the cause, be it illness of any kind, any physical or mental degradation or even an outside cause, for example, being locked in a room.

8.2 System Architecture

The instantiated unified framework is presented in Figure 8.1. First, raw sensor readings are obtained from the environment at each time step. Next, the activity recognition pipeline (ARPipe) is deployed as described in Chapter 3. It prepocesses observation vectors to reduce noise, and compute additional features. Next, an activity recognition algorithm classifies the observation vector at time t into one of the activities an individual can perform; for example, *walking, sitting, lying.* The output is a behavior trace consisting of activities and the places where they were performed. The trace is converted to a spatio-activity matrix, which is afterwards reduced with principal value decomposition. Finally, the LOF algorithm compares the behavior to an historical behavior signature database.

8.2.1 Sensors and Observations

We deployed the system in a lab organized as a home apartment, equipped with the Ubisense localization system (Steggles and Gschwind, 2009), which allows local positioning by tracking a set of tags attached to a person. The tags were placed at the following locations on the body, as shown in Figure 8.2: chest, waist, left and right ankle. Each observation vector consists of the absolute x, y, and z coordinates. The observation sequence passed to the next level was a movement trajectory of all the tags in time interval $1 \leq t \leq T$:

$$\mathbf{X} = \{[x_{tag}, y_{tag}, z_{tag}, \cdots]_t | tag \in \{chest, waist, l\_ankle, r\_ankle\}, 1 \leq t \leq T\}.$$

8.2.2 Activity Recognition Pipeline

The raw sensor data are further processed with ARPipe as described in Chapter 3. First, a median filter is applied to remove the impulsive noise. Next, an iterative constrain satisfaction method that enforces human body constraints between the measured tag positions is applied. Finally, Kalman's filter estimates additional parameters, such as tag velocity.

Activity recognition is performed in three steps. First, we extract tag attributes, such as the z coordinates, all tag velocities, the absolute distances, and the z direction distances between all the tag pairs. Activity recognition omits x and y coordinates because, from the activity-classification point of view, the location of an activity is not important. However, the x and y coordinates are essential for any daily living pattern analysis.

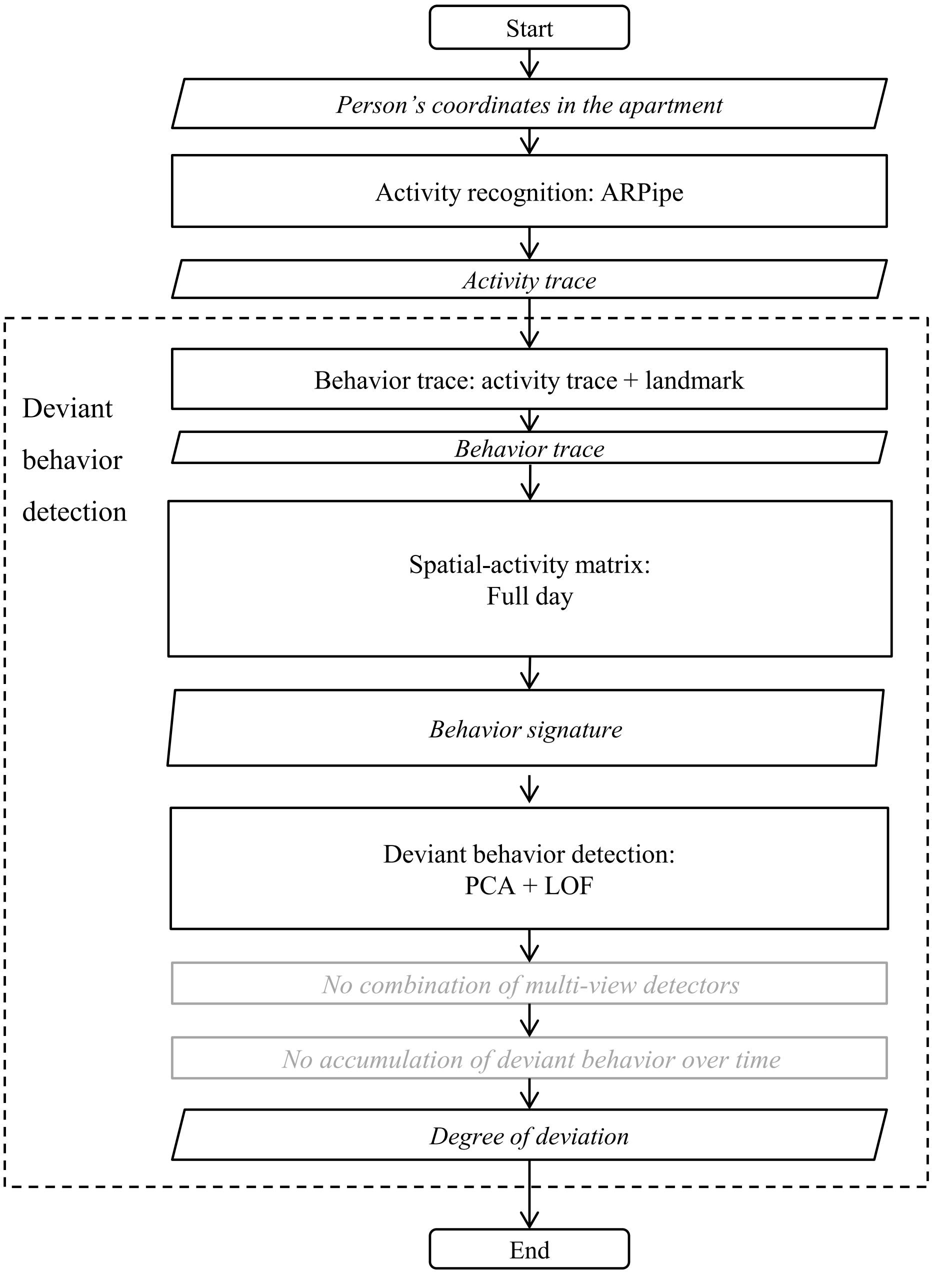

Start
Person's coordinates in the apartment
Activity recognition: ARPipe
Activity trace
Deviant behavior detection
Behavior trace: activity trace + landmark
Behavior trace
Spatial-activity matrix: Full day
Behavior signature
Deviant behavior detection: PCA + LOF
No combination of multi-view detectors
No accumulation of deviant behavior over time
Degree of deviation
End

Figure 8.1: Flowchart of the instantiated unified framework for analyzing daily-living dynamics.

Second, person postures are classified into one of the following atomic activities: $\mathbb{A} = \{walking, sitting, lying\}$. The feature vector uses canonical representation with window length $W = 10$. A new feature window is then obtained after every update, thus overlapping with the previous one, and provides instant classification for each observation vector. We have tested a variety of machine-learning algorithms (Luštrek and Kaluža, 2009), including C4.5 decision trees, naïve Bayes, SVM, k-NN, bagging, AdaBoost, etc., with SVM offering the highest classification accuracy.

Third, activity recognition errors that produced spurious activity transitions were reduced using the HMMs (Rabiner, 1989) as described in Section 3.4. Preliminary results indicated that HMMs are superior compared to sequential grammar-based classifiers (Kaluža, 2009). The HMM was initialized with $\mathbb{A} = \{walking, sitting, lying\}$ observation symbols,

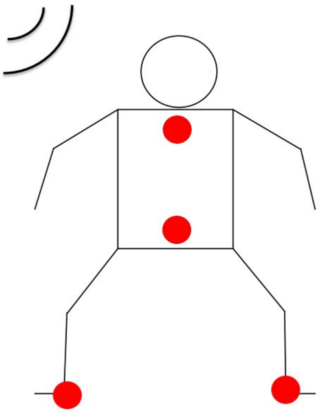


Figure 8.2: Ubisense tag placement.

three internal hidden states corresponding to activities, the initial state transition probability $\delta_{ij} = 1/3$, the initial state probability $\pi_i = 1/3$, and the output symbol distribution in state $\nu_j(k) = 1$ if $k = j$, otherwise $\nu_j(k) = 0$. The parameters were estimated with the Baum-Welch method using 50 iterations on the training data. The second phase, which finds the optimal hidden state transitions according to the observation sequence, was performed with the Viterbi algorithm.

8.2.3 Behavior Analysis

First, the behavior trace is constructed using $\langle activity, room\rangle$ tuples. The apartment was divided into logical areas $\mathbb{S} = \{lounge, bedroom, kitchen, toilet\}$ and $(x_{waist}, y_{waist})_t$ coordinates were used to determine the area at time step t. Then, the behavior trace $\mathbf{b} = \{(a, s)_t | 1 \leq t \leq T\}$ was passed to the next level.

Behavior signatures were constructed using the spatio-activity matrix approach introduced in Chapter 4 using Algorithm 4.1. The matrix dimensionality was further reduced with PCA. Finally, the LOF algorithm (Breunig et al., 2000) assigned each behavior signature an outlier degree, called the local outlier factor (LOF) of a vector. Vectors with a high LOF have local densities smaller than their neighborhood and typically represent stronger outliers, unlike vectors belonging to uniform clusters that tend to have lower LOF values.

More formally, assume that $\mathcal{B}_T = \{\mathbf{b}_i | 1 \leq i \leq L\}$ is a dataset of behavior traces. First, for each behavioral trace $\mathbf{b}_i$, we compute the spatio-activity matrix $\mathbf{M}_i$ using Algorithm 4.1. Next, we compute the principal component vector $\mathbf{m}_i$ (Equations 4.6–4.9), and add vector $\mathbf{m}_i$ to a new behavior signature dataset $\mathcal{B}$. Next, for each vector $\mathbf{m}_i$, we compute the k_dist_i as the distance to the k^{th} nearest neighbor of $\mathbf{m}_i$, then compute the reachability distance for each vector $\mathbf{m}_i$ with respect to the vector $\mathbf{m}_j$, where $d(\mathbf{m}_i, \mathbf{m}_j)$ is the Euclidean distance from $\mathbf{m}_i$ to $\mathbf{m}_j$, and compute the local reachability density lrd_i of the vector $\mathbf{m}_i$ as the inverse of the average reachability distance based on the k nearest neighbors of the vector $\mathbf{m}_i$. Finally, we compute the LOF_i of the vector $\mathbf{m}_i$ as the ratio of the average local reachability density of $\mathbf{m}_i$'s k nearest neighbors and the local reachability density of the vector $\mathbf{m}_i$.

8.3 Experimental Evaluation

For the prototype deployment we organized a room as an apartment of about 25 m^2. The apartment was equipped with a bed, a few chairs and tables, and divided into four logical

Algorithm 8.1 Anomaly detection.

```
Require: set of behavior traces $\mathcal{B}_T = \{\mathbf{b}_i | 1 \leq i \leq L\}$, number of $k$ nearest neighbors
Ensure: outlier degree for each behavior trace $LOF_i$
  $\mathcal{B} \leftarrow \{\}$
  for $\mathbf{b}_i \in \mathcal{B}_T$ do
    $\mathbf{M}_i \leftarrow spatial\_activity\_matrix(\mathbf{b}_i)$
    $\mathbf{m}_i \leftarrow PCA(\mathbf{M}_i)$
    $\mathcal{B} \leftarrow \mathcal{B} \cup \mathbf{m}_i$
  end for
  for $\mathbf{m}_i \in \mathcal{B}$ do
    $k\_dist_i \leftarrow k\_distance(\mathbf{m}_i)$
    for $\mathbf{m}_j \in \mathcal{B}, \mathbf{m}_j \neq \mathbf{m}_i$ do
      $r\_dist_{i,j} \leftarrow max(d(\mathbf{m}_i, \mathbf{m}_j), k\_dist_j))$
    end for
    $lrd_i = \frac{k}{\sum_{\mathbf{m}_j \in kNN(\mathbf{m}_i)} r\_dist_{i,j}}$
    $LOF_i \leftarrow \frac{\frac{1}{k}\sum_{\mathbf{m}_j \in kNN(\mathbf{m}_i)} lrd_j}{lrd_i}$
  end for
```

areas: a kitchen, where a person can prepare a meal; a sleeping area; a lounge, where a person can eat a meal, watch TV, write a letter, etc.; and a toilet.

8.3.1 Activity Recognition

To build an activity recognition model, we recorded five members of our department. Each participant was recorded performing various activities in three episodes lasting approximately 15–20 minutes each. In total, there were around four hours of recordings. The scenario details are available in Kaluža et al. (2010b).

The activity recognition confusion matrix presented in Table 8.1 was obtained with a *leave-one-person-out* validation. The left-hand column shows the correct-activity label, and the top row shows the assigned label. The overall classification accuracy is 87.52%.

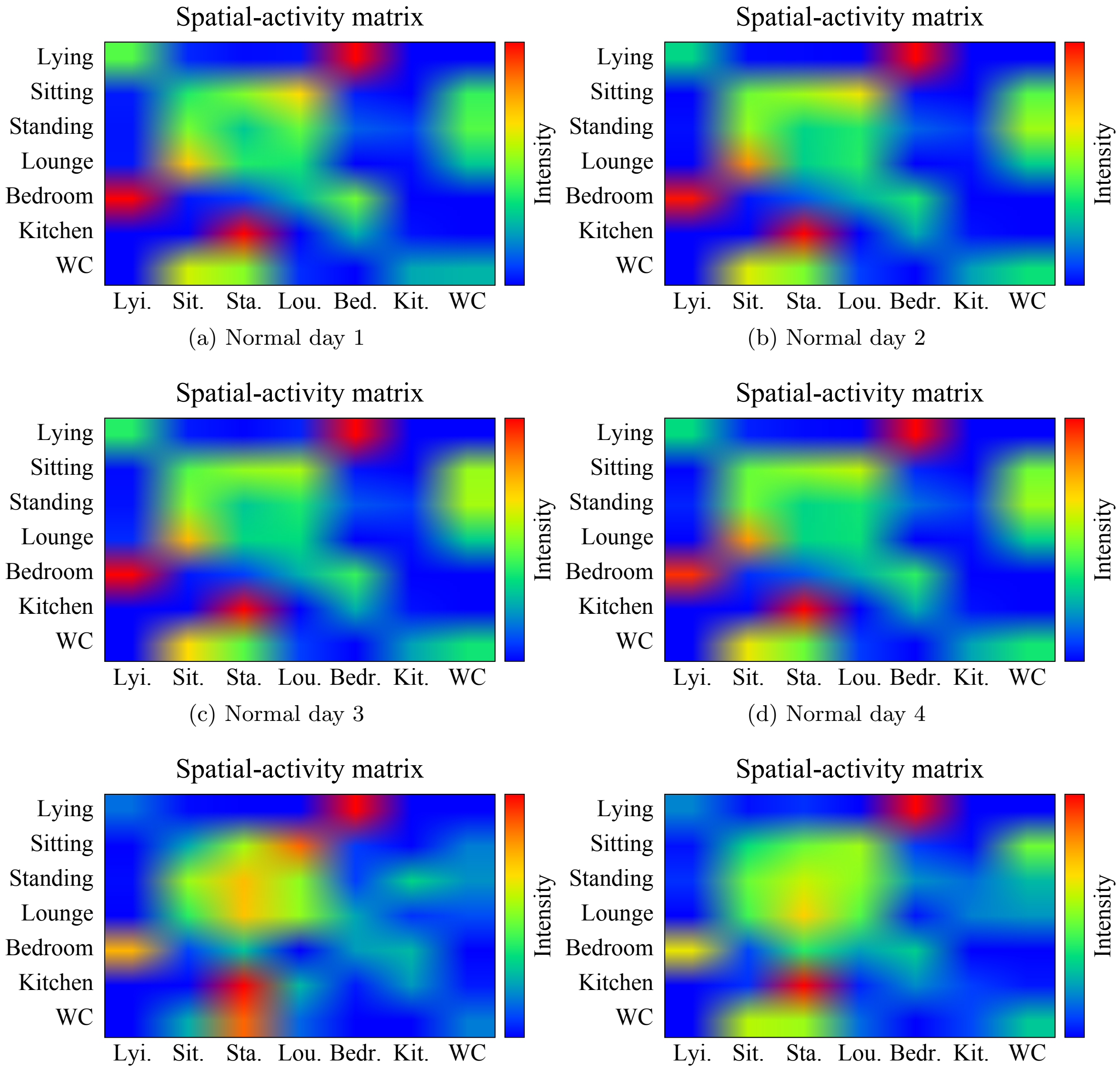
Table 8.1: Confusion matrix for activity recognition. The overall accuracy is 87.52%.

True / Labeled [%]	Lying	Sitting	Standing
Lying	98.99	0.93	0.08
Sitting	1.67	67.71	30.62
Standing	0.85	3.27	95.88

8.3.2 Anomalous Behavior Detection

We performed two experiments as follows: the first experiment condensed a full day of activities into scenarios of around half an hour each, while the second test analyzed person behavior in the office for a period of one month.

The first experiments proceeded as follows: the measurements were performed on two people aged between 25 and 32 years, with each day corresponding to a particular scenario, basically the same for each of the persons. The first, usual day represents a typical daily routine for an elderly person. It consists of sleeping, morning routine, breakfast, using toilet/household chores/reading newspaper, preparing and eating lunch, going out/watching



(a) Normal day 1

(b) Normal day 2

(c) Normal day 3

(d) Normal day 4

(e) Slow day

(f) Limping day

Figure 8.3: Behavior matrix visualization for four normal (a,b, c, d) and two deviant days (e, f) of one person.

TV/household chores/resting, dinner, watching TV/reading, and sleeping. In the second, slow day, the scenario is that the person is not feeling well and, as a consequence, is moving slowly and rests a lot. Such behavior could occur if person had the flu, heart failure, or several other general health problems, either physical or mental. In the third scenario, the person is limping due to, for example, hip pain. As a consequence, the person is also moving slowly and does not stand a lot. The person is not lying as much as on the previous day, but sits more than usual. Each person was given a loose daily scenario and an approximate timing for each activity, but performed it on her/his own. The scenarios were performed and recorded 12 times in total, consisting of eight normal days and four days where the person was not healthy. The length of the recordings varied between 25 and 40 minutes. Each recording/day was represented with one behavior trace.

First, we visually compared the usual-day scenario behavior traces with the slow-day and the limping-day scenario behavior traces. Figure 8.3 represents spatio-activity matrix

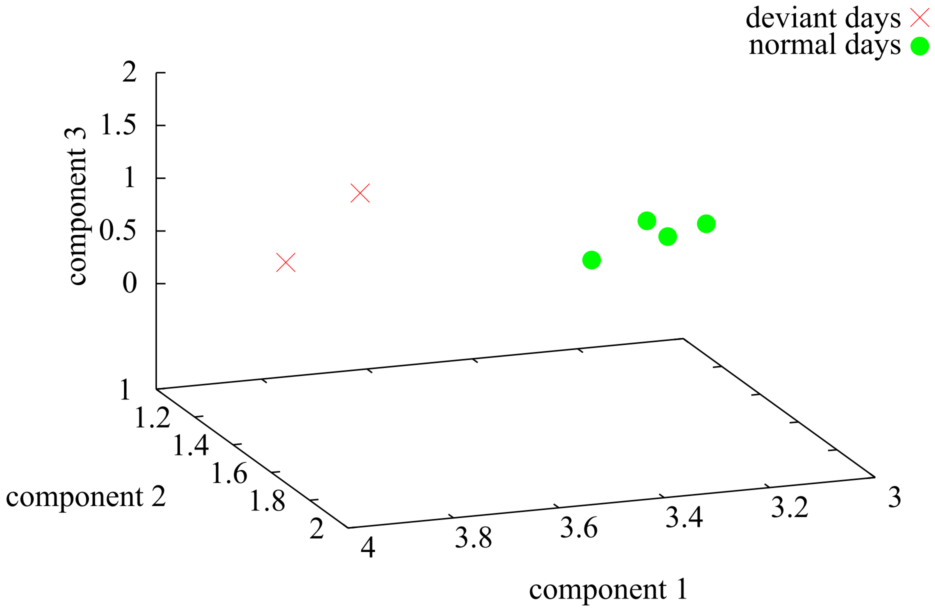


Figure 8.4: Visualization of principal components computed from the matrices shown in Figure 8.3. Normal days are presented with circles, deviation days with crosses.

visualizations computed from the behavior traces of one person for the four usual days (8.3a–8.3d) and two deviant days (8.3e, 8.3e). The spatio-activity matrices plotted in figures 8.3a–8.3d captured more or less the same daily dynamics with small variations; for example, there was slightly more standing in the toilet in day 4 (8.3d) than in day 1 (8.3a). The slow day (8.3e) had an activities over location distribution (lower-right part) quite different compared from the normal days. The most significant feature is an additional orange square, which means that there was more sitting in the lounge. The distribution also deviates during a slow day (8.3f) where, for example, the share of standing is higher than in normal days.

The difference is even more obvious when PCA is applied. Figure 8.4 shows the first three PCA components of the behavior traces plotted in Figure 8.3. The four circles '•' represent the usual days, while the other days are represented by crosses '×'.

The anomalous behavioral traces were computed using Algorithm 8.1, using *leave-one-day-out* validation; that is, one day was used for evaluation while the others were used for training. Table 8.2 shows the LOF values for different values of $k = \{2, 3\}$ for all the recordings of both persons. Normal days have $LOF < 1$ in all cases, while the anomalous days have a LOF value significantly higher than 1.

Table 8.2: LOF values of the behavior traces. A higher value represents a higher outlierness of a behavior trace.

	k=2	k=2	k=3	k=3
Scenario	User 1	User 2	User 1	User 2
Normal day 1	0.619	0.615	0.887	0.963
Normal day 2	0.694	0.613	0.904	0.766
Normal day 3	0.652	0.639	0.843	0.797
Normal day 4	0.601	0.743	0.832	0.841
Limping day	2.369	4.270	4.519	6.465
Slow day	3.274	2.358	5.451	4.227

In the second experiment, we recorded a member of our department for a period of one month. The person was recorded during working hours, approximately eight hours per day. In this experiment, the person wore only one Ubisense chest tag. The first 10 days were used for training, while the next five days were used for evaluation. Additionally, we recorded three days when the person was experiencing some difficulties: a limping day, where the person limps while he walks; an agitation day, where person occasionally walks around the office for half a minute; and an urinary tract infection day, where the person visits toilets more than usual. In total, there were 18 working days resulting in over 90 recording hours.

The results for five regular and three anomalous days are presented in Table 8.3 for $k = 2, 3, 4$. The normal working days have LOF values lower than 1 except on the third day, while the days when the person experienced some kind of difficulty, have significantly higher LOF values.

Table 8.3: LOF values of the long-term test. A higher value represents a higher outlierness of a behavior trace.

Day	k=1	k=2	k=3
Regular day 1	0.737	0.784	0.684
Regular day 2	0.803	0.698	0.594
Regular day 3	1.618	1.579	1.281
Regular day 4	0.840	0.866	0.738
Regular day 5	0.767	0.916	0.881
Limping	3.706	4.820	6.216
Agitation	7.110	8.987	12.960
Urinary infection	14.405	18.052	19.869

8.4 Discussion

In these experiments, we selected one day as a default unit, but in general, the approach can be applied to various period lengths. Furthermore, monitoring the behavior with different granularities by using different periods simultaneously; for example, half a day, a day, a week, or a month, would allow us to detect behavior changes that occur with different pace.

It should be noted that the task is based on combining activities and spatial information; therefore, applying a uniform method, such as HMMs, is not feasible. The novel method explains the two concepts by combining several existing algorithms, and specializing them for the particular task. In addition, HMMs must estimate the model learning phase parameters, whose quality depends on the labeled data amount (Rabiner, 1989).

The spatio-activity matrix visualization can be used in two ways. First, it enables visual anomalous behavior detection; by examining the matrices, one cannotice changes in behavior dynamics. Second, it explains those cases when the automatic procedure detects anomalous behavior patterns.

The final remark concerns the type of sensors used in the experiments. Even though our approach was evaluated with wireless location sensors (that is, Ubisense), it can be applied to any sensor type from which it is possible to locate and recognize activities; for example, one can use embedded sensors as shown by Cook and Holder (2011) and Storf et al. (2009).

8.5 Conclusion

The main goal of this chapter was to deliver a solution whereby a caregiver can constantly and remotely observe a person's daily behavior in an efficient and unobtrusive manner. We demonstrated how to apply the unified detection framework for transforming behavior traces into a spatio-activity matrix, which captures daily behavior and presents a visualization and explanation of behavior deviations.

We proposed a method for automatic discovery of anomalous daily behavior, which consists of feature extraction based on PCA and outlier detection implemented with the LOF algorithm. The outputs can be directly used to signal a warning to the person and caregivers, providing information that the person dynamics has changed significantly along with a relevant explanation. The experimental results showed that the proposed approach is successful in discriminating normal days from days where the person's well-being is affected.

The method has not been tested thoroughly yet. Further realistic, long-term tests with the target group are needed to verify the newly designed method's performance, and to further improve it. However, the first results are quite promising; with further modifications, the novel method for daily living dynamics might prove useful as indicated.

9 Surveillance Domain

This chapter focuses on two applications in surveillance domain, where the goal is to detect suspicious agents in the environment. In particular, the chapter targets a large applications class where no single event is sufficient to gauge whether or not agent behavior is suspicious. Instead, we face a sparse set of *trigger events* that identify interesting parts in behavior trace. The first application considers suspicious passenger detection at an airport, while the second application tackles dangerous driver detection.

9.1 Introduction and Background

There is significant suspicious activity detection research, given its importance in many domains (Arsić et al., 2007; Duong et al., 2005; Helman and Liepins, 1993; Vaswani et al., 2005). The goal is to augment the traditional security measures by scrutinizing the all subjects' behavior in the environment. The main question we address is how to combine multiple events to decide whether an event trace corresponds to the behavior of a normal or a suspicious person.

In the airport scenario, various systems were introduced to automatically detect some of the threats, such as leaving objects behind (Hongeng and Nevatia, 2003), suspicious trajectory paths (Vaswani et al., 2005), thefts (Hongeng and Nevatia, 2003), and vandalism acts and fights (Naylor and Attwood, 2003). There is also a commercially available system (Feris et al., 2009) that is able to detect such events as running passengers, climbing over a fence, etc. However, these approaches mainly deal with the detecting single, clearly suspicious incidents and do not address accumulating suspicion.

This chapter addresses how to instantiate the unified detection framework to detect trigger events, that is, interesting trace parts that serve as evidence, and combine evidence from multiple events in order to estimate suspicion. The experimental evaluation of a simulated airport application first compares the three detectors from Chapter 6 (Section 6.3) with our proposed approach (Section 6.4.1). The best two approaches are additionally compared in the dangerous-driver application.

9.2 System Architecture

Airports require numerous security solutions, including suspicious activity identification among passengers and staff in surrounding areas. Our goal is to monitor passengers and to detect those that indicate a high level of stress, fear, or deception. It is reasonable to assume that there is a camera network to track a passenger throughout the airport. We focus on a task where no single event is sufficient to identify a suspicious passenger, but a series of events establishes the decision over time. The event detection might be limited due to noise or an inability to extract some features, for example, ceiling-mounted cameras can extract passenger trajectories, but not facial expressions; hence, a normal person may appear suspicious and vice versa. Other domains may include identifying a reckless driver executing dangerous (but still legal) maneuvers (Avrahami-Zilberbrand, 2009), detecting a pirate vessel that plans to capture a transport vessel and therefore avoids security patrols, etc.

The unified detection framework is instantiated as shown in Figure 9.1. The lowest level implements a simulated environment that provides Cartesian coordinates of agents' movements in time. Atomic activity recognition is simplified to recognize relative movements from the current position, while compound activity recognition extracts trigger events; that is, interesting behavior parts. Behavior is then evaluated with two detectors and accumulated over time with approaches presented in Chapter 6.

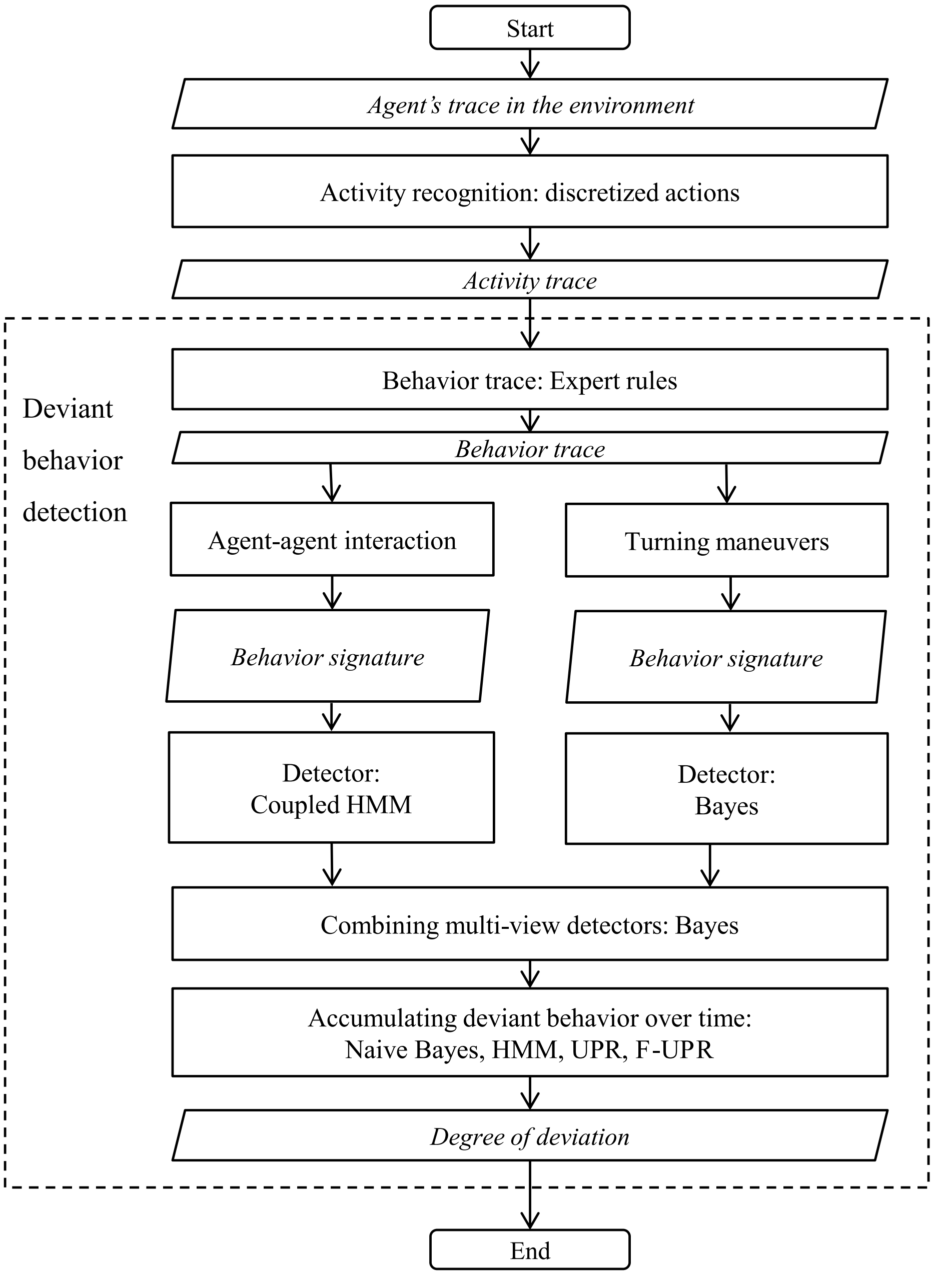

Start
Agent's trace in the environment
Activity recognition: discretized actions
Activity trace
Deviant behavior detection
Behavior trace: Expert rules
Behavior trace
Agent-agent interaction
Turning maneuvers
Behavior signature
Behavior signature
Detector: Coupled HMM
Detector: Bayes
Combining multi-view detectors: Bayes
Accumulating deviant behavior over time: Naive Bayes, HMM, UPR, F-UPR
Degree of deviation
End

Figure 9.1: Flowchart of the instantiated unified framework using two trigger event detectors and accumulating suspicious behavior.

9.2.1 Sensors and Observations

The experiments in this chapter use ESCAPES (Tsai et al., 2011), a state-of-the-art, multi-agent airport evacuations simulator with several agent types exhibiting behaviors of regular travelers, authorities, and families. The agents' behavior incorporates emotional and informational interactions, such as emotional contagion, the spread of knowledge/fear, social comparison, etc. Therefore, an agent is affected other agents' emotional states, and is faced with uncertainty as to what happened and where the nearest exits are. We assume that the agent behavior corresponds to real airport passengers behavior.

ESCAPES consists of two parts: a two-dimensional environment based on the open-source project OpenSteer (OpenSteer, 2011), outputting agents' physical trace coordinates; and a three-dimensional visualization component using the Massive Software (Regelous, 2011) to generate three-dimensional movies of the scenarios. We used a scenario that implemented the Tom Bradley International Terminal at Los Angeles International Airport, including terminals and shops as a realistic simulation environment.

In addition to behaviors already modeled within ESCAPES, we introduced a suspicious behavior profile: an agent that behaved suspiciously prior an evacuation. Additional implementation details are given in the experimetal section. An output example is shown in Figure 9.2, where traces of authorities (green), suspicious (red) and usual passengers (grey, blue) are plotted.

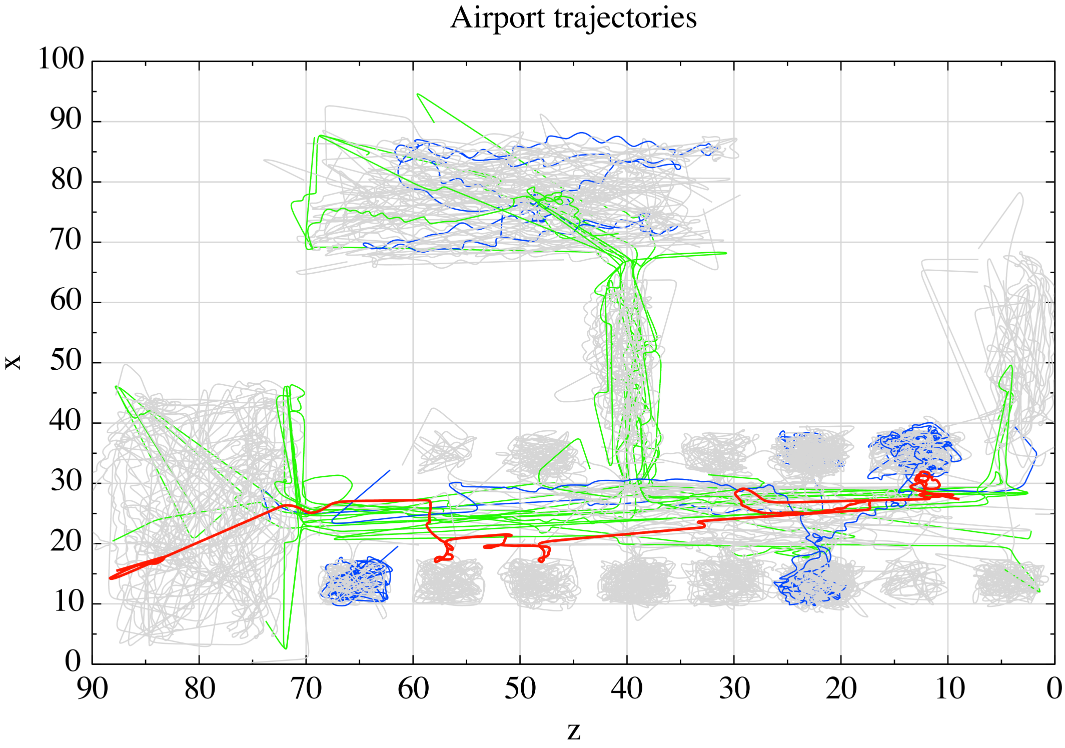


Figure 9.2: Traces of all agents at the end of a simulation: authorities (green), suspicious (red), usual passengers (grey), and selected highlighted usual passengers (blue).

An observation vector corresponds to absolute Cartesian coordinates at the airport map $\mathbf{x}_t = \langle x_t, y_t \rangle$. Observation is then a sequence of vectors obtained during simulation; that is, $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_T\}$.

9.2.2 Atomic Activities

We considered three transformations of observation vectors to actions. The first divides the airport map with a square-based grid into N numbered squares (Avrahami-Zilberbrand, 2009), which gives a set of possible atomic actions $\mathbb{A}_{fixed} = \{a_{s1}, \cdots, a_{sN}\}$. Each observation sequence with Cartesian coordinates is transformed into a sequence of squares. We denote this as *fixed representation*. By adjusting the square size, we can relax the model. By decreasing the square size, the model is more strict, less generalized, and may over-fit, since a too large size would cause over-generalization, that is, trajectories that are not similar might fit.

The second representation, denoted as *relative representation*, transforms Cartesian coordinates to actions taken in each time step as moving North, South, East, West and their combinations (nine in total); that is,

$$\mathbb{A}_{rrep} = \{a_N, a_S, a_E, a_W, a_{NE}, a_{NW}, a_{SE}, a_{SW}, a_0\}.$$

Compared to fixed representation, relative representation also describes trajectory shape, but discards the location information, which leads to better generalization.

The third representation, denoted as *relative position and orientation*, defines actions as moving Forward, Backward, Left, Right and their combinations; that is,

$$\mathbb{A}_{rrepor} = \{a_F, a_B, a_L, a_R, a_{FL}, a_{FR}, a_{BL}, a_{BR}, a_0\}.$$

Compared to relative representation, it also discards orientation information.

Preliminary tests showed the relative representation performed best (Kaluža et al., 2011e). The output of this level is an action sequence $\mathbf{a} = \{\mathbf{a}_1, \ldots, \mathbf{a}_T\}$, where an action is assigned to each observation vector.

9.2.3 Compound Activities and Agent-Agent Interactions

We focus on a well-known detector obtained from conversations with domain experts, and which is commonly used by behavior detection officers[1]. We observe the interactions between airport agents; more precisely, we are interested in how a passenger behaves in a uniformed authority figure's presence. A person exposed to a high level of stress produces behavior that indicates fear, anxiety, pressure, tension, deception, etc. Hence, it is rational for the suspicious agent to minimize contact with the authorities. Note, that no single avoidance is enough to raise a flag, but many such events taken together label the person as suspicious.

We consider two types of trigger events or compound activities: interactions with authorities $\mathbb{I} = \{pass, avoid\}$, and changes of direction $\mathbb{B} = \{turn, no\_turn\}$. The idea behind these two trigger event types is to detect the likelihood that a passenger is trying to avoid an authority figure and that the change of direction occurs in general.

Both types of compound activities are detected with basic rules. The change of direction is detected by a threshold value over a trajectory curvature, while the interaction is detected by a threshold value over the distance between an authority and a passenger. The output of this level is behavior trace, where a tuple consists of a compound activity (indicating activity) and corresponding action subsequence (indicating spatial information); that is, $\mathbf{b} = \{\langle b_1, \mathbf{a}_{b_1}\rangle, \ldots, \langle b_{T'}, \mathbf{a}_{b_{T'}}\rangle\}$. Such tuple is denoted *trigger event* e, hence $\mathbf{b} = \{e_1, \ldots, e_{T'}\}$.

[1] An approach for monitoring passengers behavior over longer periods of time relies upon security personnel such as behavior detection officers (BDOs) that patrol airports to identify passengers who display *involuntary physical and physiological actions*. US Transportation Security Administration (www.tsa.gov) trained and deployed BDO officers at over 160 US airports by 2011.

9.2.4 Suspicious Behavior Evaluation

Behavior evaluation first evaluates each trigger event and estimates the likelihood that it was generated by a legitimate or a suspicious passenger, then combines the evaluations as shown in Chapter 6.

The probabilities for each type of the trigger events are estimated separately. The turn event uses a *frequentist estimator*; that is, *a-priori* probability that a turn is generated by a suspicious or legitimate passenger:

$$\hat{n}(e_t) = \Pr\{y = 0|b = \text{turn}\}, \tag{9.1}$$

$$\hat{s}(e_t) = \Pr\{y = 1|b = \text{turn}\}. \tag{9.2}$$

Interaction probability estimation is implemented with coupled hidden Markov models (CHMMs, introduced in Section 3.6). The observations are constructed from two action sequences, namely the agent of interest's action sequence and the authority agent's action sequence when they are within some predefined radius. The CHMMs use two HMM chains, where the hidden states from one chain directly impact the hidden states from the other chain. For example, if the authority agent moves toward the suspicious agent, the next state of the latter takes this into account and produces an action for an avoidance maneuver.

A regular passenger may not turn or do anything different in the presence of authorities, while a suspicious person will, although as described below, an observer may not be perfectly observable. Therefore, we create and train two CHMMs: $\hat{N}_I$ models authorities' and regular passengers' interactions, while $\hat{S}_I$ models authorities' and suspicious passengers' interactions. For a new event (that is, interaction) e, we compute the posterior probability that the event is generated with both models yielding

$$\hat{n}(e_t) = \Pr\{e|\hat{N}\}, \tag{9.3}$$

$$\hat{s}(e_t) = \Pr\{e|\hat{S}\}. \tag{9.4}$$

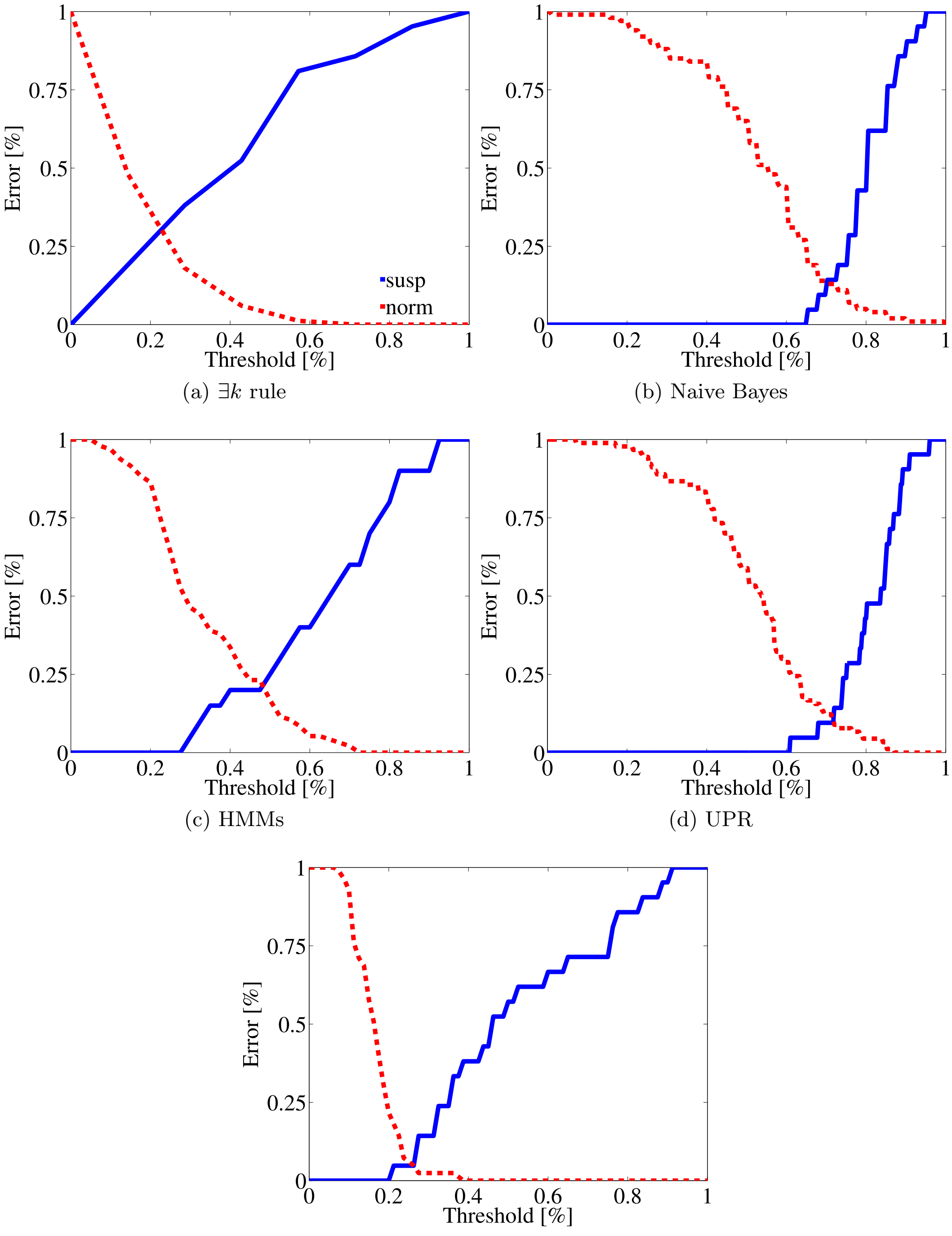
We also experimented with more complex CHMM structures including other features such as relative speed and distance, but the results were comparable or even worse.

The second step, which evaluates the trigger event sequence, uses one of the detectors introduced in Chapter 6: naïve Bayes, HMMs, UPR, and F-UPR detector.

9.3 Experimental Evaluation

In cooperation with security officials, we defined a scenario where a suspicious passenger goes from point A to point B while trying to avoid security personnel at the airport. One may argue that an adversary that plans to do something malicious would behave normally in the presence of authorities, and this might be true for a highly trained individual. As discussed previously, an average person exposed to a high level of stress exhibits fear, anxiety, and tension, and, hence, tries to cover it by minimizing close-range interactions by making U-turns, avoidance maneuvers, hiding in nearby shops, etc. The agent behavior implementation details within ESCAPES are provided in Appendix B.

A simulation in ESCAPES is run with a given airport map, authority agents, regular passengers, and a suspicious agent going from point A to point B, outputting traces with 2D coordinates for all agents. We initialized the simulator with 100 agents, including $K_a \in \{5, 10, 15, 20, 25\}$ authorities and a suspicious person with randomly chosen initial and final points. For each K_a setting, we ran 30 simulations, each consisting of 1,500–3,000 time steps and 100 traces. On average, there were 215 interactions between the authorities and the passengers per run. To avoid issues that can arise with highly unbalanced datasets, we used random re-sampling without replacement to balance the data to the ratio *suspicious : normal* = 20 : 80. For the evaluation, we used *precision*, *recall*, *specificity* and *F-measure*. We evaluate the statistical significance of our results using the two-sample t-test.



(a) $\exists k$ rule

(b) Naive Bayes

(c) HMMs

(d) UPR

(e) F-UPR

Figure 9.3: Confusion error rates for different threshold values.

Table 9.1: Evaluation results when the F-measure is maximized (columns 2–4) and all the suspicious cases are discovered (the last two columns).

	maximized F-measure			$recall$=1	
Algorithm	re	pr	FM	$1 - spec$	pr
$\exists k$ rule	0.619	0.464	0.530	1.000	0.202
Naive Bayes	0.857	0.581	0.693	0.270	0.436
HMMs	0.600	0.706	0.649	0.526	0.286
UPR	0.857	0.720	0.783	0.256	0.477
F-UPR	0.905	0.905	0.905	0.217	0.539

In the first experiment, we fixed the number of authority figures $K_a = 10$. We instantiated the naïve Bayes, HMMs, UPR, and F-UPR detectors. Additionally, we considered another baseline detector using a simple rule over the threshold k and the event trace $\mathbf{e}^{(t)}$, saying that if the number of suspicious events exceeds k (that is, $\exists k : \eta_s(\mathbf{e}^{(t)}) > k$), then mark trace $\mathbf{e}^{(t)}$ as suspicious. All the detectors used the event-trace probabilities $s'(\mathbf{e}^{(t)})$ and $n' = 1 - s'(\mathbf{e}^{(t)})$ as returned by the event-detection step. For the HMM approach, we considered two ergodic HMMs as described in Section 9.3.1. We used two observations, the normal $\Delta(e_t) = 0$ and the suspicious $\Delta(e_t) = 1$ event, and varied the hidden state number. The best results were achieved with three hidden states. Note that the HMM detector applied on top of the CHMM detector basically presents a version of the mixed-layer HMM structure (Fine et al., 1998; Nguyen et al., 2005; Duong et al., 2005). All the models, including UPR and F-UPR detectors, were evaluated with 10-fold-cross validation.

Figures 9.3(a)–9.3(e) show the confusion error rates for suspicious (1-*recall*) and normal (1-*specificity*) passengers as a function of the normalized threshold value for all the five algorithms. For example, if the threshold is zero, then all the passengers are marked as suspicious. In this case, all the suspicious passengers are correctly identified as suspicious; hence the error rate is also zero. Also, all the normal passengers are incorrectly identified as suspicious; hence the error rate is 1. As the threshold value increases, the error rate for correctly identifying suspicious passengers increases, while the error rate for correctly identifying normal passengers decreases.

There are two points of interest: (i) when the error rates cross each other; that is, the *F-measure* is maximized; and (ii) the right-most point when the error rate for suspicious passengers is zero; that is, *recall*= 1 and the false-positive rate is minimized. These cases are tabulated in Table 9.1. The first case is summarized in columns 2–4 showing the recall, precision and F-measure. F-UPR outperforms the $\exists k$ rule ($p < 0.01$), naïve Bayes ($p < 0.01$), HMMs ($p < 0.01$), and UPR ($p < 0.01$). The second case, where the threshold value is such that all the suspicious passengers are discovered, is shown in columns 5 and 6. Column 5 shows the confusion error for normal passengers (that is, 1-*specificity*), while column 6 shows the ratio of correctly raised alarms (that is, *precision*). The $\exists k$ rule, for instance, marks all the passengers as suspicious (FP rate is 100%) and consequently almost 80% of alarms are false. HMMs achieve better performance, but still mark more than 50% of normal passengers as suspicious. Other methods mark between 1/5 and 1/4 of normal passengers as suspicious, but precision is around 50%, which means that every second passenger marked as suspicious is indeed suspicious (and all suspicious passengers are discovered!). Overall, F-UPR in this setting outperforms the $\exists k$ rule ($p < 0.01$), naïve Bayes ($p < 0.05$), HMMs ($p < 0.01$), and UPR ($p < 0.05$). Finally, Figure 9.4 depicts the ROC curves showing that F-UPR performs the same, or better, in all the threshold settings.

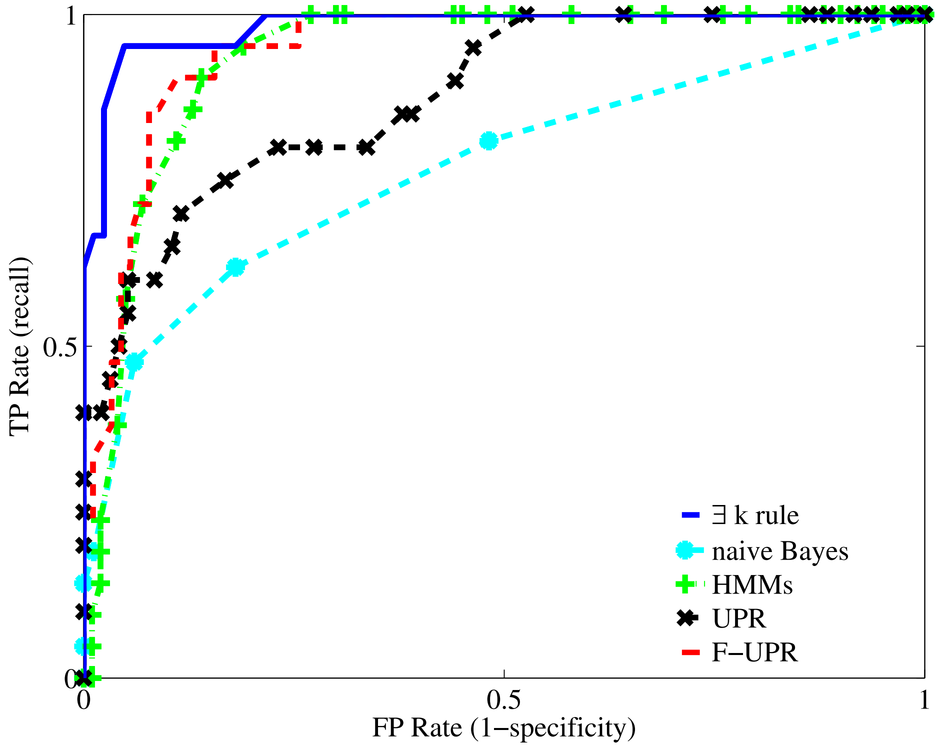


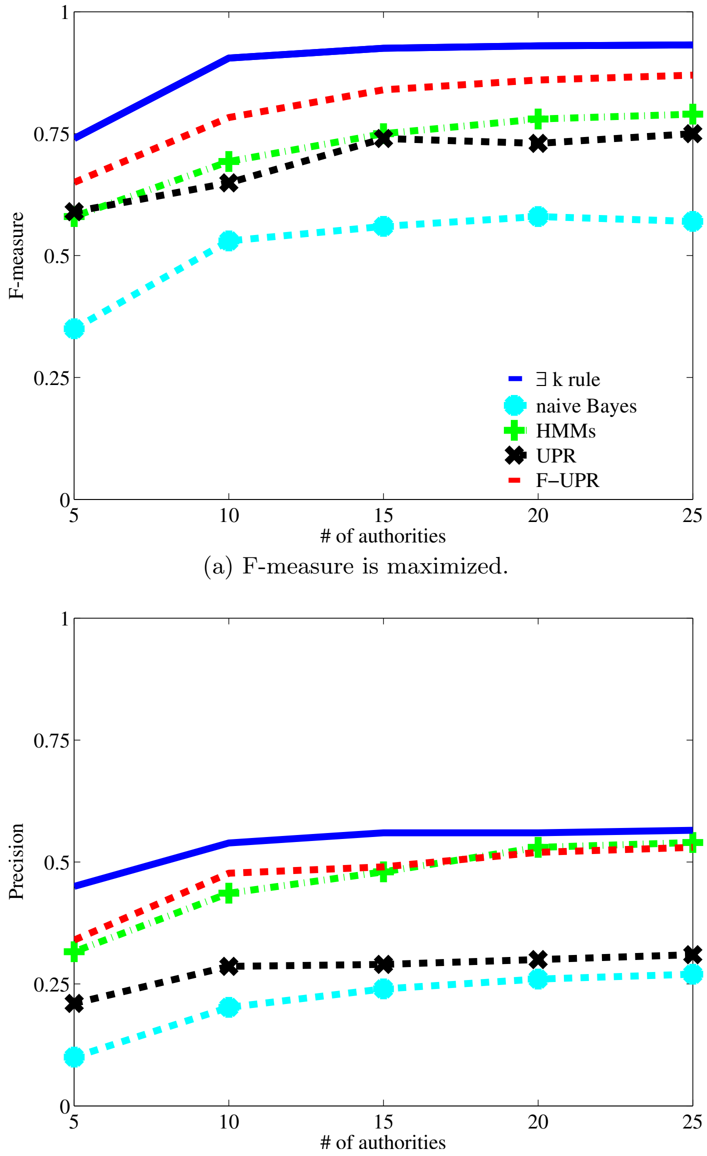
Figure 9.4: ROC curves comparing all the detectors.

In the last experiment, we varied the number of simulated authorities. We expected that increased authority figures will result in more interactions with suspicious passengers, making detection easier. Figure 9.5 shows the results for the $K_a \in \{5, 10, 15, 20, 25\}$ authority figures in a simulation: Figure 9.5a shows the F-measure for a maximizing threshold, while Figure 9.5b shows the precision when recall = 1. Increased authority figures significantly increases detection capabilities. For example, the F-measure for F-UPR increases by 15% when the security resources are doubled from five to ten, but as the number increases, the impact is smaller. We can also see that F-UPR achieves the same performance as other methods using significantly fewer security resources.

9.3.1 Detection Based on the Action Sequence

We also applied a sanity check and tested the suspicious behavior detection from a sequence of agent actions (that is, action sequence $\mathbf{a}$) instead of a sequence of trigger events (that is, event trace $\mathbf{e}$). We used HMMs, since they are considered a baseline for modeling action sequences. The goal is to differentiate between a sequence produced by a suspicious and by a regular passenger. We expect this approach to perform poorly, since it is too general to precisely model interactive behavior present in a multiagent environment.

The suspicious behavior detector consists of two ergodic HMMs: S' trained on the suspicious traces and N' trained on the regular action traces. A new trace is first transformed to the action trace $\mathbf{a}^{(k)}$ as previously described and then matched against both HMMs, yielding the likelihood that it produced the given $\mathbf{a}^{(k)}$. If the likelihood is greater than a threshold, the action trace is marked as suspicious. We tested this approach for $K_a = 10$ authorities. At the threshold value such that the highest F-measure 18.01 was achieved, this approach achieved an acceptable discovery rate (recall = 66.23) and an extremely low precision (10.42). Such a performance positions this approach under the $\exists k$ rule. The



(a) F-measure is maximized.

(b) All suspicious passengers are discovered.

Figure 9.5: Evaluation results for varying the authority figure number in the simulation and two different threshold values.

overall performance was consistent with our expectation that modeling single-agent actions in a multiagent environment would not capture the interactive behavior.

9.4 Identifying a Dangerous Driver

In addition to the airport domain, we applied UPR and F-UPR to the dangerous-driver domain, as introduced in Avrahami-Zilberbrand (2009). This domain also includes behavior that becomes increasingly costly if repeated: a driver switching a lane once or twice is not necessarily acting suspiciously, but a driver zigzagging across two lanes is dangerous. Our goal was to detect such drivers as soon as possible.

We generated 100 zigzagging driver observation sequences, each consisting of N observations, and 1,000 safe driver sequences. The trajectory observations were sampled with 10% noise. If the driver stayed on the same lane as in the previous sample, the event was

Table 9.2: Evaluation results at the peak F-measure in the dangerous driver domain.

Sequence length N	F-UPR	UPR
25	0.632	0.540
50	0.720	0.667
75	0.900	0.800
100	0.952	0.857
125	1.000	0.947

considered normal; otherwise, it was considered dangerous. For each sequence of trigger events, we accumulated the associated cost using both UPR and F-UPR. Due to observation noise, the task is expected to be more difficult when less observations are available. As the number of available observations increases, it should be easier to distinguish between safe and dangerous drivers.

Table 9.2 reports the performance at the peak F-measure for different observation sequence lengths. The results confirm the airport domain experiments for two points. First, F-UPR performs better than UPR for any selected sequence length. Second, the performance of both methods increases as the number of observations increases, where F-UPR requires fewer observations than UPR to achieve the same performance.

9.5 Conclusion

This chapter instantiated the unified detection framework to successfully address the problem of suspicious behavior detection from a set of observations, where no single observation suffices to make the decision. The chapter addressed the problem in two steps; that is, the trigger event detection and a combination of evidence to reach a final conclusion. The proposed F-UPR approach was compared to competing approaches with comprehensive experiments on two simulated domains.

With automatic behavior surveillance is it possible to identify passengers showing suspicious behavior that currently remains unnoticed. However, there are still shortcomings that are hard to bypass. For instance, observers can perceive whether someone appears anxious or is acting deceptively, they cannot tell whether that person is planning an attack or an extramarital affair. Although the *intelligent behavioral surveillance* presents an important leap in security, it raises several privacy violation concerns, which should be addressed before deploying such systems in practice.

10 Security Domain

Entry control is an important security measure that prevents undesired persons from entering secure areas. The unified detection framework utilized in this chapter allows an advanced risk analysis to distinguish between acceptable and unacceptable entries, based on several entry sensors, such as fingerprint readers, and intelligent methods that learn behavior from previous entries. First, it analyzes person behavior from different viewpoints and then performs a joint risk analysis. The obtained results represent an improvement in detecting security attacks.

10.1 Introduction and Background

Building and system safety and integrity have become more important due to the increased threat of terrorist attacks, system intrusions, and frauds. An important security requirement is to ensure effective entry controls that prevent unauthorized persons from accessing specific areas.

The general approach is to combine a two-stage security check: the identification stage, where the person introduces his/her identity; and the verification stage, based on a password and/or one or more signals derived from physical traits, such as fingerprint, voice, iris or written signature. Although widely used, entry control has certain weaknesses in the real world. Classic security methods fail to recognize unauthorized access if, for example, an identification card is stolen, a fingerprint is faked, or an employee is forced to open the door for unauthorized persons. However, intelligent access-control systems offer the promise of improved performance at a reasonable cost.

A common practice in most reported studies is to improve the two-stage security by (i) using advanced biometric methods (Wahyudi and Syazilawati, 2007; Wong and Ho, 2009; Sun and Tien, 2008); (ii) analyzing behavior (Zhang et al., 2007; Lin et al., 2009; Quah and Sriganesh, 2008; Alexandre, 1997; Wilson, 2006; Stephen and Petropoulakis, 2005; Depren et al., 2005); or (iii) combining multiple sensors into a single, reliable estimation (Lamborn and Williams, 2006; Bontempi and Borgne, 2005; Fierrez-Aguilar et al., 2005). In all the above-referenced studies, the methods successfully reduced the risk of intrusion, although each approach was focused on one specific viewpoint. Wahyudi and Syazilawati (2007), for example, presented a verification based on speech analysis. They constructed voice-based models for authorized persons and performed the identification with an adaptive network-based fuzzy-inference system. In a similar way, Wong and Ho (2009) and Sun and Tien (2008) focused on face recognition. Various facial features were extracted from video, saved in a database and compared with a new entry. The authors report an accuracy of over 90%.

Recent research efforts have focused on meta-learning (Brazdil et al., 2009; Vilalta and Drissi, 2002; Wang, 1997). The basic objective is to consider various aspects and hypotheses about an event and the environment to construct a situational awareness; then, on this basis, risk is reliably estimated. Lamborn and Williams (2006) introduced an intelligent system that consists of several heterogeneous sensors divided into clusters according to their GPS location using self-organizing maps. Sensor outputs are classified into each cluster and a voting algorithm is used to compute the final classification. Several data-mining methods were tested for cluster classification; for example, k-NNs, neural networks, and SVMs. A similar system was presented by Bontempi and Borgne (2005). In addition, Fierrez-Aguilar et al. (2005) exploited person-specific multimodal biometric parameters. They proposed an adapted local learning scheme (person-dependent) and global learning scheme (person-independent), with both results fused with weighted voting. The authors reported that the adapted learning outperformed the results from single learning.

The described approaches use state-of-the-art methods that successfully reduce intrusion risk. They use additional biometric sensors and behavior analyses as upgrades to classic access control. Our approach is a further step in combining an arbitrary number of methods in three stages. In the first stage, an arbitrary number of intelligent modules is utilized, with analyzing person behavior from different viewpoints and performing its own risk analysis. Similar to Lamborn and Williams (2006), our system constructs a situational awareness from different sensors, but, in contrast to their method, the intelligent module outputs in the second stage are assembled using meta-learning, on top of which the final reasoning is performed with a Bayesian network. In addition, the intelligent modules utilize both person-specific parameters and global knowledge similar to Fierrez-Aguilar et al. (2005), but the last integration is fused proficiently. Finally, the system is also able to explain the evaluations to a human operator and helps him/her to understand the situation. The basic assumptions of our approach are that (i) person behavior rarely changes significantly over time, and (ii) combined methods are much harder to bypass than a single sensor or method.

10.2 Hierarchal Multimodal Framework

The aim of our system is to ensure increased security in critical areas, such as military headquarters or political institutions, by detecting irregular accesses or unusual access point behavior, and, on this basis, raising an alarm. In order to reduce intrusion risk, we have designed a modular system that relies heavily on intelligent methods.

10.2.1 Functional Description

The entry procedure is shown in Figure 10.1 and is as follows: first, a person is identified. Next, if his/her identity exists, the person is verified, which leads to the door lock being released. The verification process is performed in two stages: a classic biometric verification, and an intelligent verification. Intelligent modules evaluate the entry and suggest the proper action.

The proposed intelligent access-control system's development was based on the following five requirements: first, the system must monitor entries and process evaluations in real time. Second, several access points may need to be monitored at the same time, taking into account knowledge of the person's movement between them. Third, an arbitrary number of sensors and intelligent modules will be used, depending on the equipment at specific access points and data availability. Fourth, the system is expected to evaluate an entry and suggest the proper action. Finally, the system should explain its evaluation in a user-friendly, interactive control panel. In short, the aim is to create a system that will improve entry control security and help the operator to control numerous access points effectively.

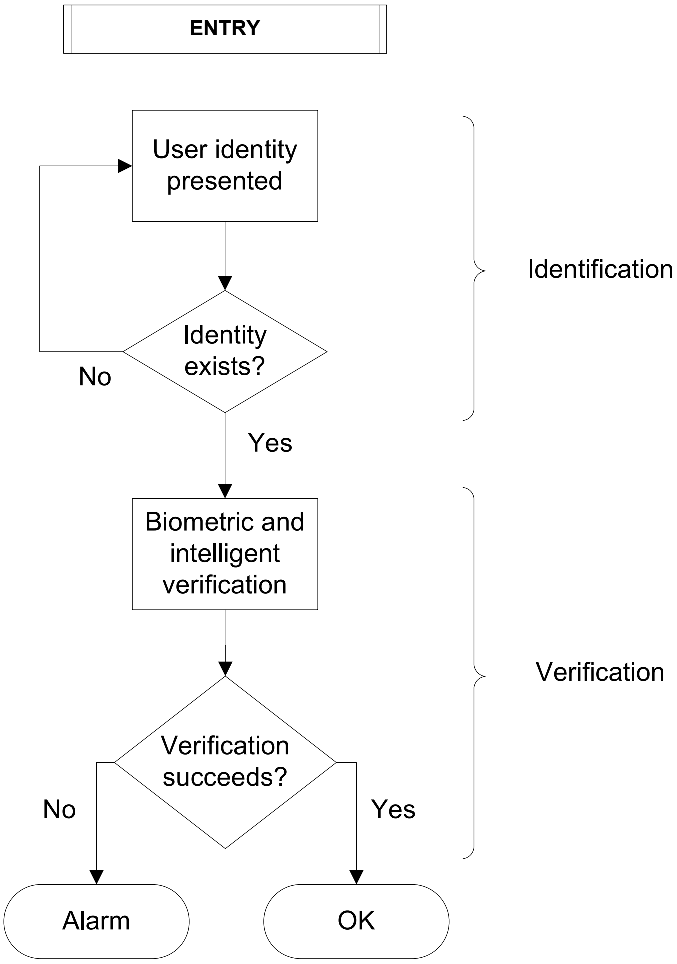


Figure 10.1: Entry identification and verification procedure at a high-secured access point.

10.2.2 Architecture

The main architectural tasks are collecting the data from the peripheral devices and sensors, processing and analyzing this data, integrating the analyses into a human-readable form, and displaying them to a person with a suggestion for an appropriate action (Figure 10.2).

The system's architecture is designed in eight layers. In the first layer, various access point sensors are deployed, such as biometric sensors, visual sensors, or door opened/closed sensors. The sensors' output is captured in the next layer through a controller, providing an observation vector $\mathbf{x}_t$ at time step t, augmented with additional contextual information.

The activity recognition layer is simplified to recognizing $\mathbb{A} = \{entry, other\}$. If all the formal criteria are met, that is, the observation consists of all the required elements, then the activity is marked as *entry* and evaluated as shown in Figure 10.1; otherwise, the entry is not completed and an alarm is raised immediately.

The next three levels are implemented in several parallel instances. Each instance constructs its own behavior patterns from a specific data type, such as visual data or temporal relations, and applies an intelligent method to evaluate the behavior. The methods include decision trees, outlier detection, expert rules, computer vision and others. Finally, all previous parallel detector outputs are gathered and the system outputs the final evaluation using a Bayesian network.

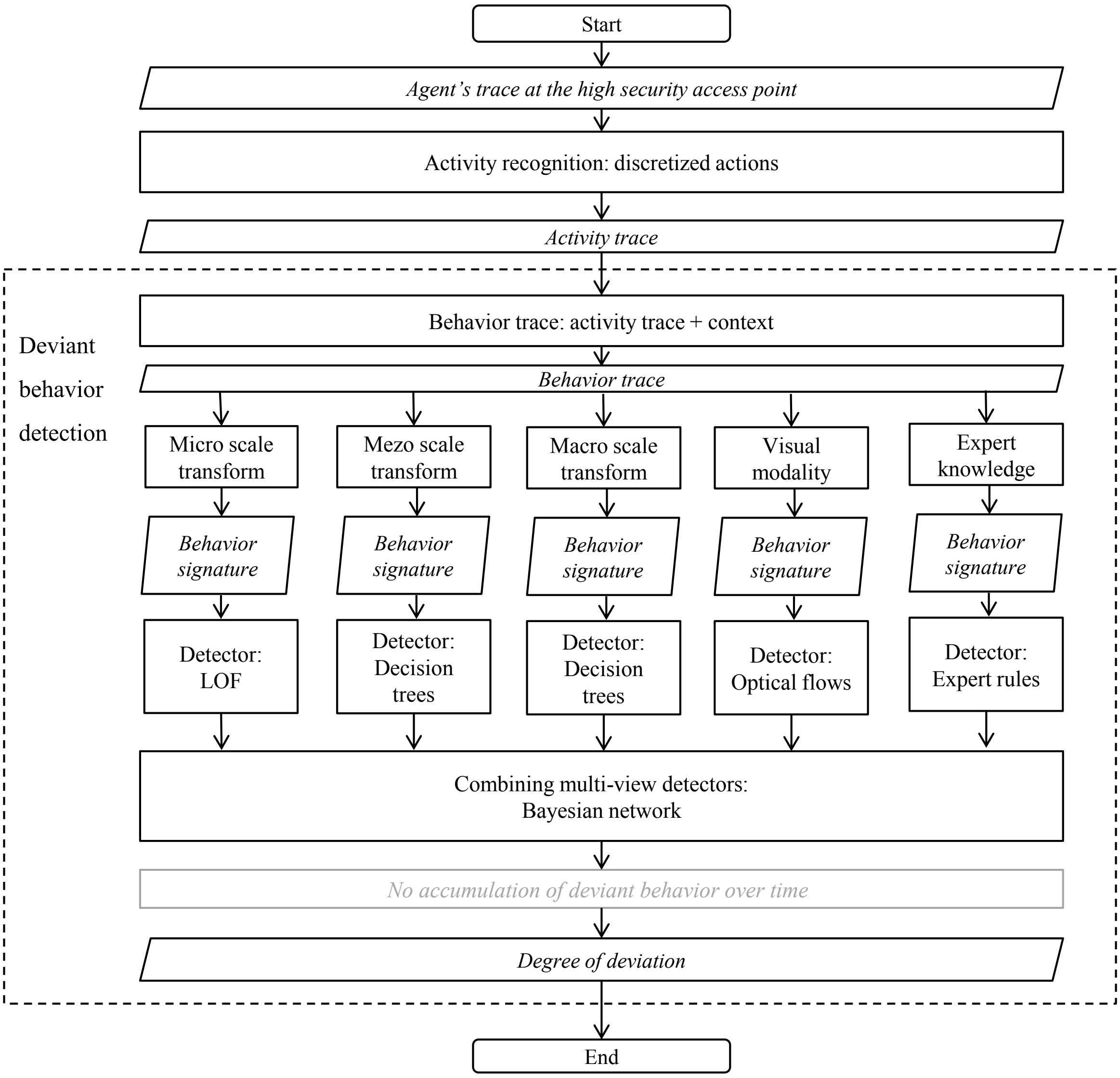


Figure 10.2: Flowchart of the instantiated unified framework using multiple time scales and modalities to evaluate behavior.

10.2.3 Observing the User's Behavior

Each human tends to perform activities in a specific way, be it on micro-or macro-scale. However, the person behavior in our system is actually monitored from three different points of view. In the first of these, denoted as the *micro-level*, one typically deals with behavior that changes in seconds or tenths of a second. For example, one person always carries his identity card in a wallet and puts the whole wallet near the wireless identity-card reader, while another person carries her card in a handbag and requires some time to take it out, identify herself, and put the card back. The person's movement around the access point depends on his/her habits and mental/physical state. These facts determine the persons' micro-level patterns.

The second viewpoint, denoted as the *macro-level*, describes the persons' daily routines. The activities are the access point arrival times, the movements between various access points in the access-control network, and even the connections between persons; for example, person A often enters a short time after person B. The time scale used at the macro-level can vary from seconds to months.

The third viewpoint, denoted as the *visual level*, captures the persons' access point visual movement using a camera. It is also focused on micro-level movements; that is, behavior that changes over a short time interval. However, in addition to micro-level features, it obtains visual characteristic features of the person and his/her movement; for example, the person's height and the door-opening dynamics.

Several rules additionally control the regular entry procedure, the regular working time, and access permissions.

10.2.4 Experimental Environment

To design and test our intelligent access control modules, we set up the experimental environment shown in Figure 10.3, which consisted of a single access point protecting an office in a building. The access point was equipped with a camera (on the ceiling), a card reader and a fingerprint reader (on the wall near the door), an electronic lock, and an open/close sensor on the door. The input signals were collected with a multi-channel access controller connected to various peripheral devices.

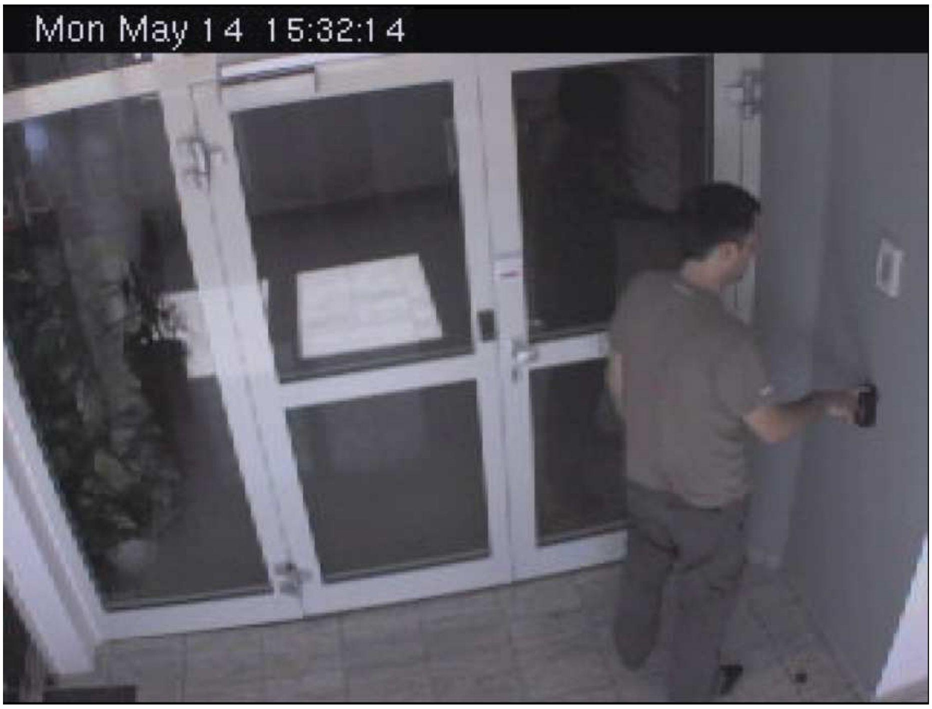


Figure 10.3: Prototype access-point configuration (camera view). The task is to detect suspicious entries of persons, for example, under the influence of drugs or with a gun that is outside the camera's field of view.

When a person passed the access point, four different times were registered:

- t_c – time of card-reader acceptance,
- t_f – time of fingerprint-reader acceptance,
- t_{do} – time of door opening,
- t_{dc} – time of door closing.

The data was collected and written into the ontology for additional processing by six intelligent modules. The first module, denoted as the *expert rules*, detected prohibited and basic undesired behavior. It used SWRL rules to query the system ontology (see Section 10.3.1). The second module, *micro-learning*, learned person micro-level behavior patterns during the entry. The learning was performed with a local outlier-detection method (LOF) (described in more detail in Section 10.3.2). The three *macro-learning* modules learned the macro-level access patterns and were then combined at the meta-level (see Section 10.3.3). The last module, *visual learning*, used optical flow histograms to detect visual-level behavior patterns (see Section 10.3.4).

Each module performed its own entry risk analysis and then returned an evaluation with an explanation. The meta-module used basic weighted voting based on single-module decisions, while the integration module accepted the module classifications as observations and performed the reasoning with a Bayesian network.

Based on the final probability, the entry was classified into one of the classes: *OK*, for regular entry, and *alarm*, for irregular entry. The system ontology stored each module's evaluations and explanations. The platform is presented in Figure 10.4.

10.2.5 Ontology

The modules and methods use the same or similar data while processing, and therefore require a comprehensible presentation. Besides the basic relationships between pieces of information, such as the sensor's value, complex representations are also required; for example, a sensor *belongs to* an access point.

We have developed an ontology using the Web Ontology Language (OWL) (Horrocks et al., 2003) and the ontology editor Protégé (2009). The ontology consists of a central part, including event data and its classifications, and several local parts, each of them storing particular module knowledge. The central part includes information about:

- access points: position, security requirements etc;
- persons: personal details, position in a company, rooms of the building that a person has permission to enter etc;
- sensors: type; for example, biometric sensor, sensor access point;
- events: person who produced the event, access point where it was produced, event sensors, individual module and final classification, and actions enabled via evaluation.

The ontology structure ensures knowledge of the system and its setting in a flexible presentation. This means that new sensors, modules, and access points can be easily added.

10.3 Detectors

This section describes the modules and algorithms in more detail. In this particular implementation, we prefer algorithms that can provide as much of an explanation as possible, but in general, it is possible to select any learning algorithm.

10.3.1 Expert Rules Detector

The first module consists of expert rules defined by a security expert or a human operator. These rules do not learn from past person behavior; instead each rule has adjustable parameters, enabling new rule creation by specifying the rule-parameter values. The rules

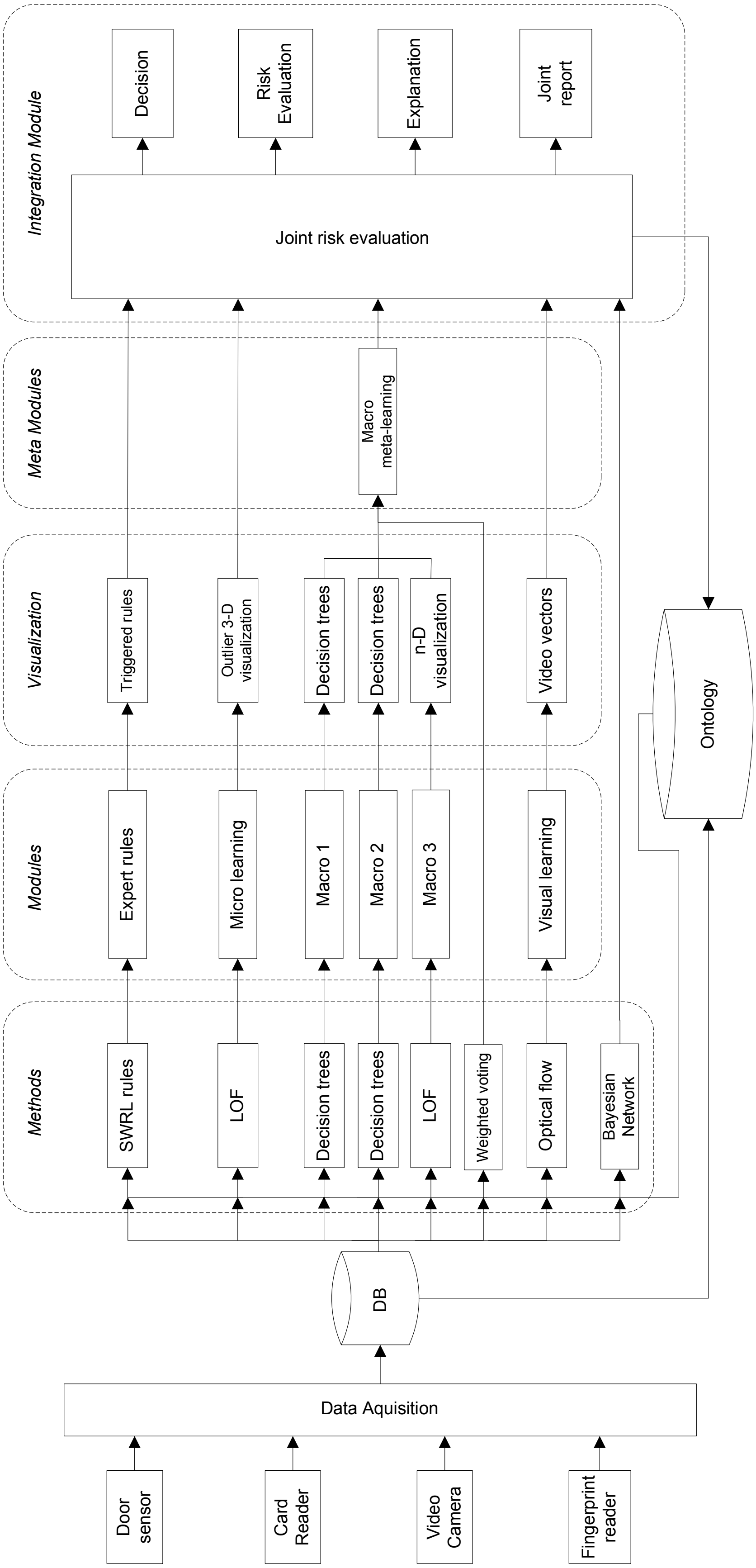


Figure 10.4: Information flow in the implemented platform.

```
event(?event_object) & swrl_end_of_testing(?event_object, ?event_swrl) &
swrlb:equal(?event_swrl, false) & card_time(?event_object, ?time_of_event) &
swrlb:greaterThan(?time_of_event,  "18:00:00") &
swrlb:lessThan(?time_of_event, "7:00:00")
THEN
swrl_rules_result(?event_object, "0.0") &
swrl_rules_explanation(?event_object, ?event_swrl_explanation) &
swrlb:stringConcat(?event_swrl_explanation,
                   "Alarm: event time is between 18:00 and 7:00")
```

Figure 10.5: An expert-rule example written in SWRL.

are described in the SWRL language (Horrocks et al., 2009) for querying data stored in the OWL. A test over the events is performed by the Jess rule engine (Friedman-Hill, 2009).

We have implemented two types of rules. If the entry procedure is violated, the first rule type triggers an alarm independently of the other modules. The second rule type refers to the entry observation; for example, *"The person accessed this area more than five times in the past two minutes"*. Instead of unconditionally triggering an alarm, each triggered rule R_i returns a probability $\Pr\{R_i\}$ that the entry is regular. If several second-type rules $R_1, \ldots, R_n$ are triggered, then $min(\Pr\{R_1\}, \ldots, \Pr\{R_n\})$ is returned and the module composes an explanation consisting of the violated rules and their parameters. Otherwise, if none of the rules is violated, the entry is regular according to the rules, and, therefore, the returned probability p equals 1.

An example of the second-type SWRL rule is shown in Figure 10.5. The rule queries events that occurred between 6:00pm and 7:00am and marks these events as alarms, since events are not allowed at night.

10.3.2 Micro-Movement Detector

The micro-learning module learns short-term behavior. The attributes are calculated as three time differences from four input times:

$$\Delta t_1 = t_f - t_c, \tag{10.1}$$

$$\Delta t_2 = t_{do} - t_f, \tag{10.2}$$

$$\Delta t_3 = t_{dc} - t_{do}. \tag{10.3}$$

Each observation $\mathbf{x}_i$ is thus represented by a triple $\mathbf{x}_i = (\Delta t_{i,1}, \Delta t_{i,2}, \Delta t_{i,3})$. All the regular entries of a particular person form a learning set $\mathcal{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$. When the person produces a new observation $\mathbf{x}_{n+k}$, the module compares it with the learning set $\mathcal{X}$ and returns an outlier factor: if the new observation is similar to the existing observations, $\mathbf{x}_{n+k}$ is a regular observation with a low outlier factor; otherwise, it is an outlier with a high outlier factor.

In previous work, Tušar and Gams (2006) examined various outlier detection algorithms, selected the LOF (Local Outlier Factor, (Breunig, 2001)) and implemented it. The algorithm reportedly achieves reliable performance where instances are not uniformly distributed in the attribute space. The LOF for a new observation $\mathbf{x}_i$ is defined as

$$LOF_k(\mathbf{x}_i) = \frac{1}{|ngb_k(\mathbf{x}_i, \mathcal{X})|} * \sum_{\mathbf{y} \in ngb_k(\mathbf{x}_i, \mathcal{X})} \frac{ldns_k(\mathbf{y})}{ldns_k(\mathbf{x}_i)}, \tag{10.4}$$

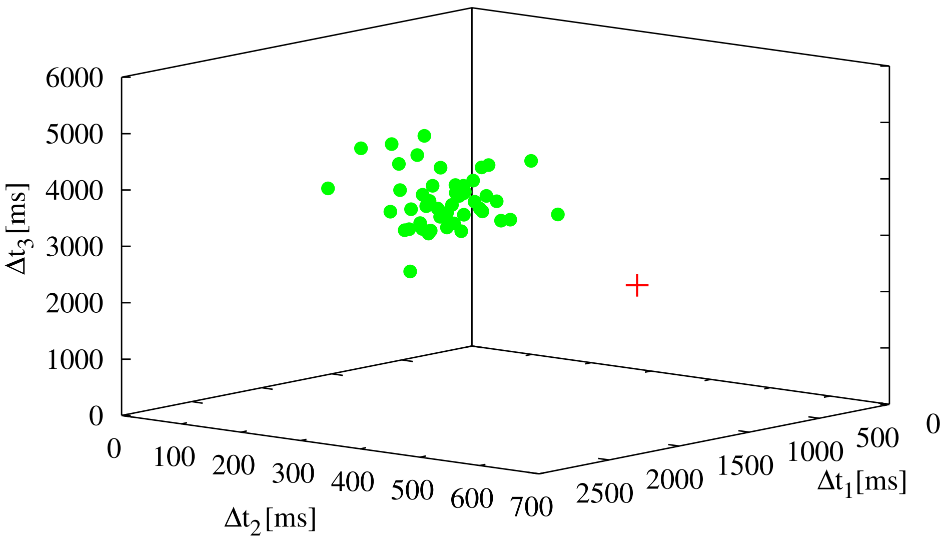


Figure 10.6: Regular entries of a particular person (circles) and a new entry denoted as an outlier ('+').

where $ngb(\mathbf{x}_i, \mathcal{X})$ is the set of k-nearest neighbors of the observation $\mathbf{x}_i$, and $ldns_k(\mathbf{y})$ is the local density of an observation $\mathbf{y}$ and its k nearest neighbors. Intuitively, $LOF_k(\mathbf{x}_i) \leq 1$ when the new observation is near an existing cluster within $\mathcal{X}$, and $LOF_k(\mathbf{x}_i) > 1$ when the observation is far from the cluster.

The final outputs of the module are the LOF value, the probability that the entry is regular, and a visual explanation. The probability is computed from the LOF value using the following procedure. Let $\tau_l < 1$ denote the regular entry threshold value and let $\tau_u > 1$ denote the irregular entry threshold value. Then, the probability $\Pr\{\mathbf{x}\}$ that the entry $\mathbf{x}$ is regular is computed as a linear threshold values combination:

$$p(e) = \begin{cases} 1.0 & \text{if } LOF(e) \leq \tau_l, \\ 0.0 & \text{if } LOF(e) \geq \tau_u, \\ \frac{\tau_u - LOF(e)}{\tau_u - \tau_l} & \text{otherwise.} \end{cases} \tag{10.5}$$

Since the module uses only three micro-attributes, its visualization can be presented in a three-dimensional space, with one dimension for each attribute. The entries are thus presented as points, and the each point's LOF value is represented by a color: red for outliers, yellow for unclear entries, and green for regular clustered entries. Figure 10.6 shows an entry cluster in a learning set $\mathcal{X}$ (circles) and a new entry $\mathbf{x}_i$ (a plus).

10.3.3 Macro-Movement and Meta-Detector

The macro-level data are used in three modules, two of which also exploit the micro-level data. The macro-level attributes are divided into two groups describing a current entry and the relation between the current entry and previous entries. The attributes from the first group are, for example, the current time and date, the day of the week, the date in relation to the month (that is, the second Friday in the month). The second group defines such relations as the number of previous entries in the same day (for the current person), the person who entered previously in a specific time interval, the entry time on the same day in the previous week, etc. It is important to note that macro-learning would be more powerful if we had monitored more than one access point.

The first macro-module learns only from macro-attributes. The positive learning examples are a person's regular entries, while the negative learning examples are a person's irregular entries and the entries of other persons. Several machine-learning algorithms were tested and, finally, decision trees were selected, Weka's J48 implementation of C4.5, in particular (Witten and Frank, 2005). The main decision tree benefit is the ability to explain a decision after classification. The path leading from the root to the chosen leaf is colored according to the classification: green for regular entries and red for alarms. Target variable distribution in the chosen leaf is interpreted as the probability that the entry is regular. The classification problem was introduced as a verification, where each person has his/her own decision tree with two possible outcomes: true, if the claimed identity is valid, and false otherwise.

The second macro-module applies the same algorithm as the previous module, but uses both micro- and macro-attributes. While the first macro-module considers only macro-level behavior and discovers patterns, for example, *"User X comes to work on Mondays between 8.15 and 8.40 (93%)"*, the second macro-module refines these patterns by incorporating micro-attributes.

In the third macro-module, the macro- and micro-attributes are used for learning with the LOF algorithm. In contrast to the micro-module, where the visualization was intuitive, the large number of attributes requires a different representation. For this purpose, we implemented parallel coordinate visualization. Each attribute is presented on one vertical axis, ranging from the minimum to the maximum normalized value. Thus, each entry is represented as a broken line intersecting the attribute value coordinates. The line is colored according to the entry's LOF value: green for regular entries, yellow for unclear entries, and red otherwise. Figure 10.7 shows a cluster of learning set entries and the new entry as a dotted line.

Finally, the macro-meta-module combines the classifications of all three macro-modules. Then, all the results and visualizations are written into the ontology. Also, in the tested implementation, only the macro-meta-learning was applied, but in principle, an arbitrary subgroup of modules could be connected using meta-learners.

10.3.4 Visual Detector

The visual learning module developed by Perš et al. (2007) learns person movement patterns using an access point video camera and classifies a new entry as either regular or not. For this purpose, a web camera with a 1.3 Mpixel resolution and 30-fps rate was used.

When a new entry occurs, the last 30 seconds of video are analyzed in the following steps: first, the optical flow histograms are computed and divided into six segments, approximating body parts. Next, in each segment, the prevailing movement is estimated and transferred into a sequence of symbols. This sequence defines the movement's digital signature and is used for the verification. Each person has a learning set of valid regular entries, which are compared with the new entry signatures. Finally, the module outputs the classification and probability that the entry is regular as a normalized comparison.

It should be noted that other sensor analyses, such as speech or walking patterns, could be added as well.

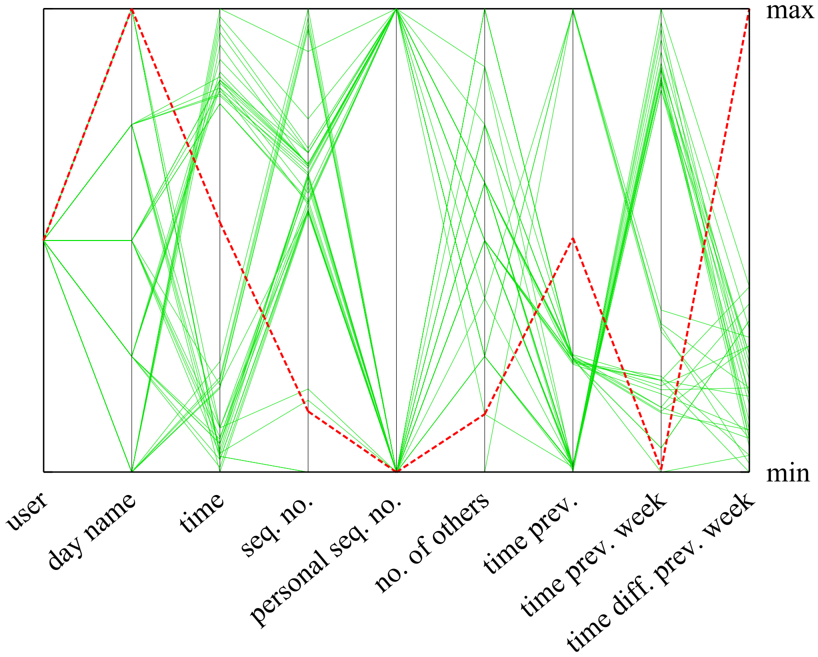


Figure 10.7: Multi-dimensional representation of regular entries (thin lines) and a new entry (dotted line) classified as an alarm. There are nine attributes with values normalized between the minimum and maximum values.

10.3.5 Multimodal Detector Integration

After the expert rules, micro-learning, macro-learning, meta-learning, and visual-learning have made their assessments, their results are integrated into a current entry joint risk analysis. It estimates the event probability $\Pr\{\textit{entry} \mid \textit{entry is regular}\}$ given the module observations. If the estimated probability does not exceed a threshold value, an alarm is triggered. Note that an alarm can also be triggered by expert rules when there is sufficient certainty.

The reasoning in the prototype system is performed with a Bayesian network, structured as shown in Figure 10.8. Four modules have a direct impact on the entry event; that is, expert rules, micro-learning and visual learning, and a macro-meta-learning module, while the macro-meta-learning module depends only on the three macro-modules. The network probabilities are computed from the train dataset, using the *m-estimate* for conditional probabilities and the *Laplace estimate* for a-priori probabilities.

The integration proceeds in three steps. Firstly, the output from each module is converted to interval the $[0,1]$ representing the a-posteriori probability $\Pr\{M_i\}$ that the entry event is regular. Secondly, given the Bayesian network and the probabilities $\Pr\{M_i\}$, the estimated probability of an entry event is computed from the network.

Finally, the integration module outputs the joint analysis as a probability that the entry is regular and provides an explanation. According to the threshold values, the integration module triggers *alarm* or *OK* and stores the results in the ontology. In high-security areas, the cost of a false alarm is negligible compared to the cost of an unrecognized intruder; therefore, the system is set to minimize the latter.

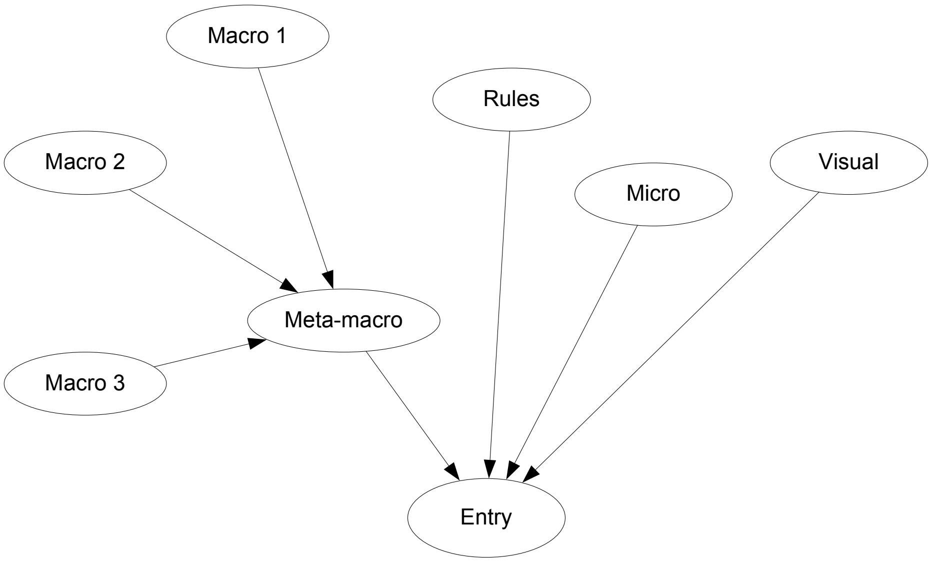


Figure 10.8: Bayesian network used for reasoning.

10.4 Experimental Results

An experimental verification was performed in the prototype environment as described in Section 10.2.4. It consisted of learning and evaluation phases. In this chapter, we report on one learning and three evaluation experiments.

10.4.1 Learning Phase

In the learning phase, four people were recorded accessing the system. Each individual completed 40 regular entries that were used as positive learning examples. The negative learning examples for one individual were the entries of the other three people. We built decision trees for the macro-modules, constructed learning sets for the LOF algorithm in the micro-and macro-module and a comparison set for the visual learning module, and adjusted the system parameters. After the learning was completed, the system was ready to operate.

10.4.2 Evaluation Phase

In the evaluation phase, we performed three experiments: two with simulated entries and one real-time experiment with security experts.

The first two experiments were performed off-line with simulated tests. The focus was on a *fake-identity* scenario, where an adversary has stolen an employee's identity. We recorded the regular entries of four people in the role of an employee (the system already knew them) and three people in the role of an intruder (new to the system). Each person made 31 regular entries, serving as the testing examples. Both experiments were tested without the visual learning since it did not allow offline testing. Consequently, the Bayesian network for the integration was slightly changed, omitting the visual learning module. The experiments were run on already-learned and tuned modules from the learning phase, while the Bayesian network probabilities were obtained with a 10-fold cross-validation.

In the first experiment, the identities of the employees were swapped. We took four employees that were known to the system and shuffled their identities in order to simulate a scenario where an employee hands over his/her identity. The dataset contained 496 examples with a distribution of 75% negative examples (fake identity).

The system and the module performance in the first experiment is presented in Table 10.1. The first two columns represent irregular entries, where employee identity was swapped, and regular entries with the correct employee identity. Each number denotes an accuracy; for example, the left-most number represents the irregular entry percentage predicted as regular by the expert rules. The last column presents the overall module accuracy. The system achieved an overall accuracy of 95.77%. The expert rules always predicted *OK*, because all the entries were formally regular according to the entry procedure. The micro-learning detected both irregular and regular entries well, while the macro-learning had 10.08% more mistakes. The high accuracy of the micro-module was expected because it is relatively easy to distinguish the movement of a couple of people given sufficient learning examples.

In the second experiment, we used the intruders' entries, which were unknown to the system, and assigned them the employees' identities. In this way, we simulated a stolen-identity scenario. The dataset consisted of 496 examples with a distribution 75% of negative examples.

The second experiment measurements are shown in Table 10.2. The system achieved an overall accuracy of 96.57%. In contrast with the results in Table 10.1, where macro-learning classified 16.13% false positives, the number of false positives in Table 10.2 is only 1.88%. However, the trend in the micro-learning is just the opposite; the overall accuracy is comparable in both datasets. The decline in micro-learning performance was to be expected, since it is more difficult to classify new, unseen behavior than to distinguish between known cases.

	Scenarios				
	Irregular entries		Regular entries		Overall
	OK	*alarm*	*OK*	*alarm*	Accuracy
Modules	[%]	[%]	[%]	[%]	[%]
Expert rules	100.00	0.00	100.00	0.00	25.00
Micro learning	5.91	94.09	92.74	7.26	93.75
Macro learning	16.13	83.87	83.06	16.94	83.67
Integration	1.08	98.92	86.29	13.71	95.77

Table 10.1: System and module performance in the offline *swapped identity* experiment with four employees only.

	Scenarios				
	Irregular entries		Regular entries		Overall
	OK	*alarm*	*OK*	*alarm*	Accuracy
Modules	[%]	[%]	[%]	[%]	[%]
Expert rules	100.00	0.00	100.00	0.00	25.00
Micro learning	22.04	77.96	92.74	7.26	81.65
Macro learning	1.88	98.12	82.26	17.74	94.15
Integration	0.00	100.00	86.29	13.71	96.57

Table 10.2: System and module performance in the offline *stolen identity* experiment with four employees and three intruders.

In the third, most relevant experiment, we invited security experts from the Slovenian Ministry of Defense to test the system with a live simulation of various security attacks.

For the purpose of scientific experimentation, the following eight scenarios were proposed, tested and executed live by the experts:

1. regular entry: a person enters normally;
2. unusual time: the access time is out of normal working hours or on a non-working day;
3. multiple entries: a person regularly accesses a secure room several times in a short period of time;
4. unusual behavior: a person is under threat or in a strange state of mind;
5. tailgating: two persons access a secure room using a single identity;
6. burglary: an attacker disables the hardware protection by force;
7. fake identity: an attacker accesses a secure room with a stolen identity card and a forged fingerprint;
8. kidnapping: an attacker forces an employee to enable secure room access.

Each scenario was imitated several times by different persons and in a different order, as requested by the security experts. In total, 45 irregular entries and 15 regular entries were performed. The video learning module was active.

The results described in Table 10.3 are separated into two groups: regular entries (scenario 1) and irregular entries (scenarios 2-8). The numbers show the percentage of test examples classified as *OK*, *alarm* or *failed* by the corresponding module. The classification may fail due to the disabling of sensors (for example, the burglary scenario).

	Scenarios					
	Irregular entries			Regular entries		Overall
	OK	*alarm*	*failed*	*OK*	*alarm*	Accuracy
Modules	[%]	[%]	[%]	[%]	[%]	[%]
Expert rules	84.44	15.56	0.00	100.00	0.00	36.67
Micro learning	0.00	88.89	11.11	93.33	6.67	90.00
Macro learning	0.00	88.89	11.11	86.67	13.33	88.33
Visual learning	8.89	88.89	4.44	73.33	26.67	85.00
Integration	0.00	100.00	0.00	86.67	13.33	96.67

Table 10.3: System and module performance in the experiments with four employees and four security experts in a role of intruder.

The system achieved an overall accuracy of 96.67%, identifying all the irregular entries, and being too suspicious of two regular entries. Once again, the expert rules classified with a low accuracy (36.67%), but when an entry was classified as an alarm, it was indeed so. The rules were more robust compared to the other modules, which, for example, failed to recognize the burglary scenario. The micro- and macro-learning modules recognized the irregular entries with the same accuracy, but macro-learning made more mistakes when classifying the regular entries. It should be noted that all the tests were performed within two hours, which is not well suited to macro-learning. The visual learning was slightly more robust (that is, less failures) than the learning modules, but achieved a lower accuracy.

10.5 Discussion

We have designed a modular, intelligent system for analyzing access point intrusion risk. The system, in principle, combines an arbitrary number of intelligent modules on top of an arbitrary number of physical devices. The emphasis is on modeling the behavior of the regular person and estimating the risk that a new entry is not regular, based on meta-learning and integration.

In a practical evaluation[1] we presented three experiments, which demonstrated encouraging results. It was clear that each module has its own strong and weak points. However, an advanced combination and integration overcomes the individual weaknesses and combines different aspects into a reliable risk evaluation. For example, if we had used only the best module (micro-learning) in the third experiment, the achieved accuracy would have been 90.00%, while the default accuracy (which is rather meaningless) would have been 75.00%. The accuracy of the integrated system was 96.67%.

In each system, there is a fine line between being too sensitive and not being sensitive enough to small changes in behavior. Although some of the methods, for example, the Bayesian network, are quite robust, any practical application needs some fine tuning of the system parameters. One of the first major benchmarks painfully reminded us of the difference between a laboratory test and a field test; that is, one of the early system versions was able to successfully distinguish between normal persons, but security experts found a way to trick the intelligent modules. Only after incorporating some modifications, was the system able to cope with human expertise, as presented in Table 10.3.

One of the system drawbacks is that it requires a learning procedure: the system can be used only after a certain amount of regular accesses have been made. Furthermore, if a person changes behavior, for example, due to an injury, the learning must start anew. Further work on the system must include a mechanism for continuous learning and person adaptation over time.

The complex methods implemented seem to be excessive for a simple commercial application. In its current form, the system is more appropriate for high-security areas. Namely, the joint-verification methods turned out to be very hard to bypass. A single method can be fooled relatively easily, but deceiving different methods within a normal time interval is a much harder task.

In summary, intelligent access-point risk analysis represents an improvement and has the potential to demonstrate this in real-time applications.

[1] A short video of the third experiment is available online: http://www.youtube.com/watch?v=BNDgfFRQkU4.

11 Conclusions

11.1 Summary and Discussion

This thesis addressed the problem of deviant behavior pattern detection within a large class of problems with complex, spatio-temporal, sequential data generated by an entity capable of physical motion in an environment.

The central scientific hypothesis of this thesis states that *it is possible to leverage the available spatio-temporal cues, temporal dependencies, various time scales and modalities, and repetitive behavior patterns to detect anomalous and suspicious behavior.*

To this end, we developed new methods to extract spatio-temporal cues and temporal dependencies, and proposed a unified detection framework to address various viewpoints, as well as repetitive behavior patterns for anomalous and suspicious behavior detection. To examine the validity of the hypothesis, we empirically demonstrated the unified detection framework on three domains. In the ambient assisted living domain, we demonstrated how to apply the framework to monitor an elderly person in a home environment to detect daily living anomalies, where the key component is an activity recognition pipeline and a spatio-activity matrix analysis. In the surveillance domain, we addressed the issue of repeated behavior detection and applied the framework to detect suspicious passengers at the airport. The novel F-UPR detector significantly outperformed the competing approaches. Finally, in the security domain, where the goal is to verify entering persons at a high-security access control point, we demonstrated a proof of concept of how the multimodal detection is beneficial. In summary, the thesis hypothesis is supported by the empirical evaluation and thus confirmed.

11.2 Scientific Contributions

The work in this thesis has led to the following original contributions to science:

1. **A unified anomalous and suspicious behavior detection framework**: We proposed a unified framework for detection of anomalous and suspicious behavior that can be observed from complex, spatio-temporal sequential data generated by an agent moving in a physical environment. The framework incorporates several components to address the main challenges and is demonstrated empirically in three studies.

2. **Contribution to anomalous and suspicious behavior detection**: We gave the first clear problem definition and established a theoretical framework for anomalous and suspicious behavior detection from agent traces to show **how to optimally perform detection**. We discussed why detection error is often inevitable and **proved the lower error bound**. We further provided several heuristic approaches that either estimated distributions required to perform detection or directly rank the behavior signatures using machine-learning approaches.

3. **Contribution to repeated behavior detection**: We extend the established theoretical framework and showed how to perform detection when an agent is observed over longer periods of time and no significant event is sufficient to reach decision. We first specified **conditions any reasonable detector should satisfy** and analyzed several detectors. We further proposed a novel approach denoted as **F-UPR detector** that generalizes utility-based plan recognition with arbitrary utility functions.

4. **Contribution to behavior analysis**: We proposed a novel, efficient encoding denoted as **spatio-activity matrix** that is able to capture behavior dynamics in a specific time period using spatio-temporal features. We provided a visualization technique to compare different behavior patterns. We further provided a feature extraction technique based on principal component analysis to reduce the spatio-activity matrix dimensionality, which can be directly used in anomaly detection algorithms.

5. **Contribution to activity recognition**: To address the problem of activity recognition from sensor data we introduced ARPipe, an **Activity Recognition Pipeline** that includes filtering, attribute construction, activity recognition, and activity smoothing. Within the pipeline, several novel algorithms were introduced including **body filter**, which applies human-body constraints to location-based body-attached sensors, and two approaches for **reducing spurious activity transitions** that cannot occur in reality are demonstrated.

11.3 Future Work

Anomalous and suspicious behavior patterns are rare, hence, a direction for future work is to consider approaches to expedite their appearance. For example, if the obtained deviation degree does not lead to confirmation, an observer might trigger an action toward the observed agent and observe its response to disambiguate his intentions, as it was demonstrated on an air-combat domain in a seminal work by Tambe and Rosenbloom (1995).

The unified framework proposed in this thesis has certain limitations in terms of deployment; for example, once the framework is trained and installed it does not take into consideration any feedback provided by the human operators behind it. One way to overcome this is to consider an online-learning mechanism that is able to incorporate human operator feedback in future behavior evaluations. Such a mechanism must not only adapt specific detectors to provide feedback, but also has to take into account gradual behavior drift of the agents interacting with(in) the environment.

12 References

Albrecht, D. W.; Zukerman, I.; Nicholson, A. E. Bayesian models for keyhole plan recognition in an adventure game. *User Modeling and User-Adapted Interaction* **8**, 5–47 (1998).

Albusac, J.; Castro-schez, J. J.; Lopez-lopez, L. M.; Vallejo, D.; Jimenez-linares, L. Learning and classification of events in monitored environments. In: *Proceedings of the Joint 2009 International Fuzzy Systems Association World Congress and 2009 European Society of Fuzzy Logic and Technology Conference.* 375–380 (EUSFLAT Press, Lisbon, Portugal, 2009).

Alexandre, T. J. Biometrics on smart cards: an approach to keyboard behavioral signature. *Future Generation Computer Systems* **13**, 19–26 (1997).

Armentano, M. G.; Amandi, A. Plan recognition for interface agents. *Artificial Intelligence Review* **28**, 131–162 (2009).

Arsić, D.; Hörnler, B.; Schuller, B.; Rigoll, G. A hierarchical approach for visual suspicious behavior detection in aircrafts. In: *Proceedings of the 16th international conference on Digital Signal Processing.* 639–645 (IEEE Computer Society Press, Piscataway, NJ, USA, 2009).

Arsić, D.; Schuller, B.; Rigoll, G. Suspicious behavior detection in public transport by fusion of low-level video descriptors. In: *Proceedings of the 8th IEEE International Conference on Multimedia and Expo.* 218–221 (IEEE Computer Society Press, Beijing, China, 2007).

Augusto, J. C.; Huch, M.; Kameas, A.; Maitland, J.; McCullagh, P. J.; Roberts, J.; Sixsmith, A.; Wichert, R. (eds.) *Handbook of Ambient Assisted Living – Technology for Healthcare, Rehabilitation and Well-being*, vol. 11 of *Ambient Intelligence and Smart Environments* (IOS Press, Amsterdam, Netherlands, 2012).

Avrahami-Zilberbrand, D. *Efficient Hybrid Algorithms for Plan Recognition and Detection of Suspicious and Anomalous Behavior* (Ph. D. thesis, Bar-Ilan University, Tel Aviv, Israel, 2009).

Avrahami-Zilberbrand, D.; Kaminka, G. A. Incorporating observer biases in keyhole plan recognition (efficiently!). In: *Proceedings of the 22nd National Conference on Artificial Intelligence.* 944–949 (AAAI Press, Vancouver, British Columbia, Canada, 2007).

Bak, P.; Rohrdantz, C.; Leifert, S.; Granacher, C.; Koch, S.; Butscher, S.; Jungk, P.; Keim, D. A. Integrative visual analytics for suspicious behavior detection. In: *IEEE Symposium on Visual Analytics Science and Technology.* 253–254 (IEEE Computer Society Press, Atlantic City, New Jersey, USA, 2009).

Banos, O.; Damas, M.; Pomares, H.; Rojas, F.; Delgado-Marquez, B.; Valenzuela, O. Human activity recognition based on a sensor weighting hierarchical classifier. *Soft Computing* **17**, 333–343 (2013).

Barbará, D.; Domeniconi, C.; Duric, Z.; Filippone, M.; Mansfield, R.; Lawson, E. Detecting suspicious behavior in surveillance images. In: *IEEE International Conference on Data Mining Workshops*. 891–900 (IEEE Computer Society Press, Omaha, NE, USA, 2008).

Baum, L. E.; Petrie, T.; Soules, G.; Weiss, N. A maximization technique occurring in the statistical analysis of probabilistic functions of Markov chains. *The Annals of Mathematical Statistics* **41**, 164–171 (1970).

Biermann, E.; Cloete, E.; Venter, L. A comparison of intrusion detection systems. *Computers and Security* **20**, 676–683 (2001).

Bontempi, G.; Borgne, L. An adaptive modular approach to the mining of sensor network data. In: *Proceedings of the Workshop on Data Mining in Sensor Networks*. 41–48 (ACM, Newport Beach, CA, 2005).

Bouchard, B.; Giroux, S.; Bouzouane, A. A keyhole plan recognition model for alzheimer's patients: First results. *Applied Artificial Intelligence* **21**, 623–658 (2007).

Bourke, A.; Lyons, G. A threshold-based fall-detection algorithm using a bi-axial gyroscope sensor. *Medical Engineering & Physics* **30**, 84–90 (2008).

Brand, M.; Oliver, N.; Pentland, A. Coupled hidden Markov models for complex action recognition. In: *Computer Vision and Pattern Recognition*. 994–999 (IEEE Computer Society Press, San Juan, Puerto Rico, 1997).

Brazdil, P.; Giraud-Carrier, C.; Soares, C.; Vilalta, R. *Metalearning: Applications to Data Mining* (Springer, Berlin, Heidelberg, Germany, 2009).

Breunig, M. *Quality Driven Database Mining* (Ph. D. thesis, University of Munich, Munich, Germany, 2001).

Breunig, M.; Kriegel, H.-P.; Ng, R. T.; Sander, J. LOF: Identifying density-based local outliers. In: *Proceedings of the 2000 ACM Sigmod International Conference on Management of Data*. 93–104 (ACM, Dallas, Texas, USA, 2000).

Cardinaux, F.; Bhowmik, D.; Abhayaratne, C.; Hawley, M. S. Video based technology for ambient assisted living: A review of the literature. *Journal of Ambient Intelligence and Smart Environments* **3**, 253–269 (2011).

Chen, L.; Hoey, J.; Nugent, C.; Cook, D.; Yu, Z. Sensor-based activity recognition. *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews* **42**, 790–808 (2012).

Chen, L.; Khalil, I. Activity recognition: Approaches, practices and trends. In: *Activity Recognition in Pervasive Intelligent Environments*. 1–31 (Atlantis Press, Amsterdam, Nederland, 2011).

Chen, L.; Özsu, M. T.; Oria, V. Robust and fast similarity search for moving object trajectories. In: *Proceedings of the 2005 ACM SIGMOD international conference on Management of data*. 491–502 (ACM, Baltimore, Maryland, USA, 2005).

Choudhury, T.; Philipose, M.; Wyatt, D.; Lester, J. Towards activity databases: Using sensors and statistical models to summarize people's lives. *IEEE Data Engineering Bulletin* **29**, 49–56 (2006).

Cook, D. J. Multi-agent smart environments. *Journal of Ambient Intelligence and Smart Environments* **1**, 51–55 (2009).

Cook, D. J.; Holder, L. B. Sensor selection to support practical use of health-monitoring smart environments. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery* **1**, 339–351 (2011).

Cvetković, B.; Kaluža, B.; Gams, M. Prilagajanje modela za razpoznavanje aktivnosti človeka z aktivnim delno nadzorovanim učenjem. In: *Proceedings of the 13th International Multiconference Information Society.* 63–66 (Jožef Stefan Institute, Ljubljana, Slovenia, 2010).

Cvetković, B.; Luštrek, M.; Kaluža, B.; Gams, M. Semi-supervised learning for adaptation of human activity recognition classifier to the user. In: *Workshop on Space, Time and Ambient Intelligence, International Joint Conferences on Artificial Intelligence (IJCAI 2011).* 24–29 (AAAI Press, Barcelona, Spain, 2011).

Cvetković, B.; Luštrek, M.; Kaluža, B.; Gams, M. Multi-classifier adaptive training: Specializing an activity recognition classifier using semi-supervised learning. In: *International Joint Conference on Ambient Intelligence*, vol. 7683 of *Lecture Notes in Computer Science.* 193–207 (Springer, Pisa, Italy, 2012).

Dee, H. M.; Hogg, D. C. On the feasibility of using a cognitive model to filter surveillance data. In: *IEEE Conference on Advanced Video and Signal Based Surveillance.* 34–39 (IEEE Computer Society Press, Milano, Italy, 2005).

Dee, H. M.; Hogg, D. C. Navigational strategies and surveillance. In: *Proceedings IEEE International Workshop on Visual Surveillance.* 73–81 (IEEE Computer Society Press, Marseilee, France, 2008).

Dee, H. M.; Hogg, D. C. Navigational strategies in behaviour modelling. *Artificial Intelligence* **173**, 329–342 (2009).

Demirdjian, D. Enforcing constraints for human body tracking. In: *Conference on Computer Vision and Pattern Recognition Workshop*, vol. 9. 102 (IEEE Computer Society Press, Madison, Wisconsin, USA, 2003).

Depren, O.; Topallar, M.; Anarim, E.; Ciliz, M. K. An intelligent intrusion detection system (IDS) for anomaly and misuse detection in computer networks. *Expert Systems with Applications* **29**, 713–722 (2005).

Dore, A.; Pinasco, M.; Ciardelli, L.; Regazzoni, C. A bio-inspired system model for interactive surveillance applications. *Journal of Ambient Intelligence and Smart Environments* **3**, 147–163 (2011).

Dovgan, E.; Cvetković, B.; Mirchevska, V.; Kaluža, B.; Luštrek, M.; Gams, M. Improving the quality of life for elderly. In: *Confidence International Conference.* 99–108 (University of Jyväskylä, Jyväskylä, Finland, 2010a).

Dovgan, E.; Kaluža, B.; Tušar, T.; Gams, M. Agent-based security system for user verification. In: *IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology.* 331–334 (IEEE Computer Society Press, Milan, Italy, 2009).

Dovgan, E.; Kaluža, B.; Tušar, T.; Gams, M. Improving user verification by implementing an agent-based security system. *Journal of Ambient Intelligence and Smart Environments* **2**, 21–30 (2010b).

Duda, R. O.; Hart, P. E.; Stork, D. G. *Pattern Classification (2nd Edition)* (Wiley-Interscience, New York, NY, USA, 2000).

Duong, T. V.; Bui, H. H.; Phung, D. Q.; Venkatesh, S. Activity recognition and abnormality detection with the switching hidden semi-Markov model. In: *Computer Vision and Pattern Recognition.* 838–845 (IEEE Computer Society, San Diego, CA, USA, 2005).

Duque, D.; Santos, H.; Cortez, P. Prediction of abnormal behaviors for intelligent video surveillance systems. In: *IEEE Symposium on Computational Intelligence and Data Mining.* 362–367 (IEEE Computer Society Press, Honolulu, Hawaii, USA, 2007).

Ektefa, M.; Memar, S.; Sidi, F.; Affendey, L. Intrusion detection using data mining techniques. In: *International Conference on Information Retrieval Knowledge Management (CAMP).* 200–203 (IEEE Computer Society Press, Shah Alam, Selangor, 2010).

Elkan, C. The foundations of cost-sensitive learning. In: *Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence.* 973–978 (Morgan Kaufmann Publishers, San Francisco, CA, USA, 2001).

eMotion. Smart Motion Capture System. http://www. emotion3d.com/smart (accessed: November 2009).

Esponda, F.; Forrest, S.; Helman, P. A formal framework for positive and negative detection schemes. *IEEE Systems Man and Cybernetics Society* **34**, 357–373 (2004).

Feris, R. S.; Hampapur, A.; Zhai, Y.; Bobbitt, R.; Brown, L.; Vaquero, D. A.; li Tian, Y.; Liu, H.; Sun, M.-T. *IBM Smart Surveillance System* (Taylor & Francis Group, London, United Kingdom, 2009).

Fierrez-Aguilar, J.; Garcia-Romero, D.; Ortega-Garcia, J.; Gonzalez-Rodriguez, J. Adapted user-dependent multimodal biometric authentication exploiting general information. *Pattern Recognition Letters* **26**, 2628–2639 (2005).

Fine, S.; Singer, Y.; Tishby, N. The hierarchical hidden Markov model: Analysis and applications. *Machine Learning* **32**, 41–62 (1998).

Fong, S. Mining suspicious patterns in physical environment. In: *IEEE International Workshop on Anti-counterfeiting, Security, Identification.* 419–422 (IEEE Computer Society Press, Xiamen, Fujian, 2007).

Fong, S.; Yan, Z. A security model for detecting suspicious patterns in physical environment. In: *Third International Symposium on Information Assurance and Security.* 221–226 (IEEE Computer Society Press, Manchester, United Kingdom, 2007).

Friedman-Hill, E. Jess, the rule engine for the java platform. http://www.jessrules.com (accessed: November 2009).

Gams, M. *Weak Intelligence: Through the Principle and Paradox of Multiple Knowledge* (Nova Science, Huntington, NY, USA, 2001).

Gams, M.; Krivec, J.; Kaluža, B.; Rode, A. Varnostni sistem CIVaBiS: celoviti inteligentni varnostni biometrični sistem. *Obramba* **41**, 30–33 (2009).

Gams, M.; Luštrek, M.; Kaluža, B. Patologija končno razložena? In: *Proceedings of the 11th International Multiconference Information Society*, vol. A. 19–21 (Jožef Stefan Institute, Ljubljana, Slovenia, 2008).

Geib, C. W.; Goldman, R. P. Extended abstract: Recognizing plan / goal abandonment. In: *AAAI Technical Report FS-02-05*. 1515–1517 (AAAI Press, North Falmouth, Massachusetts, USA, 2002).

Geib, C. W.; Goldman, R. P. A probabilistic plan recognition algorithm based on plan tree grammars. *Artificial Intelligence* **173**, 1101–1132 (2009).

Gimon, D.; Gjoreski, H.; Kaluža, B.; Gams, M. Using accelerometers to improve position-based activity recognition. In: *Proceedings of the 13th International Multiconference Information Society (IS 2010)*. 15–18 (Jožef Stefan Institute, Ljubljana, Slovenia, 2010).

Gorup, Č.; Tavčar, A.; Kaluža, B. Eavesdroping on VoIP network. In: *Proceedings of the Sixteenth International Electrotechnical and Computer Science Conference*, vol. A. 69–72 (IEEE Slovenija, Portorož, Slovenia, 2007).

Goshorn, R. *Sequential Behavior Classification Using Augmented Grammars* (Master's thesis, University of California, San Diego, California, USA, 2001).

Govindarajan, M.; Chandrasekaran, R. Intrusion detection using k-nearest neighbor. In: *International Conference on Advanced Computing*. 13–20 (IEEE Computer Society Press, Chennai, India, 2009).

Gyanchandani, M.; Rana, J. L.; Yadav, R. N. Taxonomy of anomaly based intrusion detection system: A review. *International Journal of Scientific and Research Publications* **2**, 790–808 (2012).

Helman, P.; Liepins, G. Statistical foundations of audit trail analysis for the detection of computer misuse. *IEEE Transactions on Software Engineering* **19**, 886–901 (1993).

Helman, P.; Liepins, G.; Richards, W. Foundations of intrusion detection. In: *The IEEE Computer Security Foundations Workshop V*. 114–120 (IEEE Computer Society Press, Franconia, NH, USA, 1992).

Hongeng, S.; Nevatia, R. Large-scale event detection using semi-hidden markov models. In: *IEEE International Conference on Computer Vision*. 1455–1462 (IEEE Computer Society Press, Nice, France, 2003).

Horrocks, I.; Patel-Schneider, P. F.; Boley, H.; Tabet, S.; Grosof, B.; Dean, M. SWRL: A semantic web rule language combining OWL and RuleML. http://www.w3.org/Submission/SWRL/ (accessed: November 2009).

Horrocks, I.; Patel-Schneider, P. F.; Harmelen, F. V. From SHIQ and RDF to OWL: The making of a web ontology language. *Journal of Web Semantics* **1**, 7–26 (2003).

Huỳnh, T.; Blanke, U.; Schiele, B. Scalable recognition of daily activities with wearable sensors. In: *Proceedings of the 3rd International Conference on Location-and Context-Awareness*, LoCA'07. 50–67 (Springer, Berlin, Heidelberg, 2007).

Jakobsen, T. Advanced character physics. In: *Game Developers Converence Proceedings*. 383–401 (CMP Media, Inc., San Jose, California, USA, 2001).

Kalman, R. E. A new approach to linear filtering and prediction problems. *Transactions of the ASME–Journal of Basic Engineering* **82**, 35–45 (1960).

Kaluža, B. Reducing spurious activity transitions in a sequence of movement. In: *Proceedings of the Eighteenth International Electrotechnical and Computer Science Conference*, vol. B. 163–166 (IEEE Slovenija, Portorož, Slovenia, 2009).

Kaluža, B.; Cvetković, B.; Dovgan, E.; Gjoreski, H.; Mirchevska, V.; Gams, M.; Luštrek, M. Multiagent care system to support independent living. *International Journal on Artificial Intelligence Tools,* accepted (2013).

Kaluža, B.; Dovgan, E. Glajenje trajektorij gibanja človeškega telesa zajetih z radijsko tehnologijo. In: *Proceedings of the 12th International Multiconference Information Society*, vol. A. 97–100 (Jožef Stefan Institute, Ljubljana, Slovenia, 2009).

Kaluža, B.; Dovgan, E.; Cvetković, B.; Mirchevska, V.; Luštrek, M.; Gams, M. A multiagent system for remote eldercare. In: *Workshop on Agents for Ambient Assisted Living, International Conference on Practical Applications of Agents and Multi-Agent Systems.* 33–40 (Springer, Salamanca, Spain, 2011a).

Kaluža, B.; Dovgan, E.; Luštrek, M.; Gams, M. Context aware MAS to support elderly people. In: *Eleventh International Conference on Autonomous Agents and Multiagent Systems, AAMAS 2012.* 1485–1488 (IFAAMAS, Valencia, Spain, 2012a).

Kaluža, B.; Dovgan, E.; Mirchevska, V.; Luštrek, M.; Gams, M. Intelligent monitoring of the elderly in home environment. In: *AI and Health Communication, AAAI 2011 Spring Symposium Series.* 1 (AAAI Press, Stanford, California, USA, 2011b).

Kaluža, B.; Dovgan, E.; Tušar, T.; Gams, M. Intelligent risk analysis in access control. In: *Proceedings of the International Workshop on Quantitative Risk Analysis for Security Applications in conjunction with International Joint Conferences on Artificial Intelligence (IJCAI 2009)*, vol. A. 9–16 (AAAI Press, Pasadena, California, USA, 2009a).

Kaluža, B.; Dovgan, E.; Tušar, T.; Tambe, M.; Gams, M. A probabilistic risk analysis for multimodal entry control. *Expert Systems with Applications* **38**, 6696–6704 (2011c).

Kaluža, B.; Gams, M. Odkrivanje terorističnih vstopov s pomočjo ambientalne inteligence. In: *Proceedings of the Seventeenth International Electrotechnical and Computer Science Conference*, vol. B. 145–148 (IEEE Slovenija, Portorož, Slovenia, 2007a).

Kaluža, B.; Gams, M. Razpoznavanje števk pri varnostnem geslu zvočne CAPTCHA zaščite. In: *Proceedings of the Seventeenth International Electrotechnical and Computer Science Conference*, vol. B. 371–372 (IEEE Slovenija, Portorož, Slovenia, 2007b).

Kaluža, B.; Gams, M. Intelligent access control system based on users behavior. In: *Procedings of the 2008 Networking and Electronic Commerce Research Conference.* 305–308 (ATSMA, Riva del Garda, Italy, 2008a).

Kaluža, B.; Gams, M. Z ambientalno inteligenco nad teroriste. *Življenje in tehnika* **59**, 23–29 (2008b).

Kaluža, B.; Gams, M. An approach to analysis of daily living dynamics. In: *Proceedings of the World Congress on Engineering and Computer Science 2010*, vol. 1 of *International Conference on Machine Learning and Data Analysis, ICMLDA-10.* 485–490 (Newswood Limited, San Francisco, CA, USA, 2010).

Kaluža, B.; Gams, M. Analysis of daily-living dynamics. *Journal of Ambient Intelligence and Smart Enviroments* **4**, 403–413 (2012).

Kaluža, B.; Kaminka, G.; Tambe, M. Identifying suspicious behavior from multiple events. In: *3rd Jožef Stefan International Postgraduate School Students' Conference*. 74–79 (IPS, Ljubljana, Slovenia, 2011d).

Kaluža, B.; Kaminka, G.; Tambe, M. Towards detection of suspicious behavior from multiple observations. In: *Workshop on Plan, Activity, and Intent Recognition, AAAI Conference on Artificial Intelligence (AAAI-11)*. 8–18 (AAAI Press, San Francisco, California, 2011e).

Kaluža, B.; Kaminka, G.; Tambe, M. Detection of suspicious behavior from a sparse set of multiagent interactions. In: *Eleventh International Conference on Autonomous Agents and Multiagent Systems, AAMAS 2012*. 955–964 (IFAAMAS, Valencia, Spain, 2012b).

Kaluža, B.; Kozina, S.; Luštrek, M. Activity recognition repository: Towards competitive benchmarking in AmI. In: *Workshop on Activity Context Representation, AAAI Conference on Artificial Intelligence (AAAI-12)*. 44–47 (AAAI Press, Toronto, Canada, 2012c).

Kaluža, B.; Luštrek, M. Fall detection and activity recognition methods for the Confidence project: A survey. In: *Proceedings of the 12th International Multiconference Information Society*, vol. A. 22–25 (Jožef Stefan Institute, Ljubljana, Slovenia, 2008).

Kaluža, B.; Luštrek, M.; Gams, M. Patologija minimaksa v sintetičnih drevesih in Pearlovi igri. In: *Proceedings of the 10th International Multiconference Information Society*, vol. A. 84–87 (Jožef Stefan Institute, Ljubljana, Slovenia, 2007a).

Kaluža, B.; Luštrek, M.; Gams, M.; Tavčar, A. Pathology in minimax searching. In: *Proceedings of the Sixteenth International Electrotechnical and Computer Science Conference*, vol. B. 107–110 (IEEE Slovenija, Portorož, Slovenia, 2007b).

Kaluža, B.; Mirchevska, V.; Dovgan, E. Context-aware MAS for remote elderly care. In: *Proceedings of the 2nd Jožef Stefan International Postgraduate School Students' Conference*. 26 (IPS, Ljubljana, Slovenia, 2010a).

Kaluža, B.; Mirchevska, V.; Dovgan, E.; Luštrek, M.; Gams, M. An agent-based approach to care in independent living. In: *International Joint Conference on Ambient Intelligence (AmI-2010)*, vol. 6439 of *Lecture Notes in Computer Science*. 177–186 (Springer, Malaga, Spain, 2010b).

Kaluža, B.; Mirchevska, V.; Luštrek, M.; Vélez, I.; Gams, M. Ubiquitous care system to support independent living: Preliminary results. In: *Roots for the future of ambient intelligence: Adjunct proceedings*. 308–315 (Springer, Salzburg, Austria, 2009b).

Kangas, M.; Konttila, A.; Lindgren, P.; Winblad, I.; Jamsa, T. Comparison of low-complexity fall detection algorithms for body attached accelerometers. *Gait & Posture* **28**, 285–291 (2008).

Kaye, K. Tsa screening more than just carry-on bags. *The Washington Post* (November 9, 2009).

Klingsch, W. W. F.; Rogsch, C.; Schadschneider, A.; Schreckenberg, M.; Boltes, M.; Seyfried, A.; Steffen, B. *Pedestrian and Evacuation Dynamics 2008* (Springer, Berlin, Heidelberg, 2010).

Koller, D.; Friedman, N. *Probabilistic Graphical Models: Principles and Techniques* (MIT Press, Cambridge, Massachusetts, United States, 2009).

Kruegel, C.; Mutz, D.; Robertson, W.; Valeur, F. Bayesian event classification for intrusion detection. In: *Proceedings of the 19th Annual Computer Security Applications Conference.* 14–23 (IEEE Computer Society Press, Las Vegas, NV, USA, 2003).

Kwapisz, J. R.; Weiss, G. M.; Moore, S. A. Activity recognition using cell phone accelerometers. *ACM SIGKDD Explorations Newsletter* **12**, 74–82 (2011).

Lamborn, P.; Williams, P. J. Data fusion on a distributed heterogeneous sensor network. In: *Multisensor, Multisource Information Fusion: Architectures, Algorithms, and Applications*, Proceedings SPIE. 1–8 (SPIE Press, Orlando, FL, USA, 2006).

Lane, T.; Brodley, C. E. Temporal sequence learning and data reduction for anomaly detection. *ACM Transactions on Information and System Security* **2**, 295–331 (1999).

Lee, B.; Martin, T.; Clarke, N.; Majeed, B.; Nauck, D. Dynamic daily-living patterns and association analyses in tele-care systems. In: *Fourth IEEE International Conference on Data Mining.* 447–450 (IEEE Computer Society Press, Brighton, United Kingdom, 2004).

Lee, J.; Hoff, W. Activity identification utilizing data mining techniques. In: *IEEE Workshop on Motion and Video Computing (WMVC'07).* 12 (IEEE Computer Society Press, Austin, USA, 2007).

Li, X.; Han, J.; Kim, S. Motion-alert: Automatic anomaly detection in massive moving objects. In: Mehrotra, S.; Zeng, D. D.; Chen, H.; Thuraisingham, B.; Wang, F.-Y. (eds.) *Intelligence and Security Informatics*, vol. 3975 of *Lecture Notes in Computer Science.* 166–177 (Springer, Berlin, Heidelberg, Berlin, Heidelberg, 2006).

Lin, L.; Seo, Y.; Gen, M.; Cheng, R. Unusual human behavior recognition using evolutionary technique. *Computers and Industrial Engineering* **56**, 1137–1153 (2009).

Lou, H.; Chai, J. Example-based human motion denoising. *IEEE Transactions on Visualization and Computer Graphics* **16**, 870 –879 (2010).

Luo, C.; Zhao, Y.; Cao, L.; Ou, Y.; Zhang, C. Exception mining on multiple time series in stock market. In: *IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology.* 690–693 (IEEE Computer Society Press, Sydney, Australia, 2008).

Luštrek, M.; Kaluža, B. Fall detection and activity recognition with machine learning. *Informatica* **33**, 197–204 (2009).

Luštrek, M.; Kaluža, B.; Cvetković, B.; Dovgan, E.; Gjoreski, H.; Mirchevska, V.; Gams, M. Confidence: Ubiquitous care system to support independent living. In: *Proceedings of 20th biennial European Conference on Artificial Intelligence (ECAI 2012).* 1013–1014 (IOS Press, Montpellier, France, 2012).

Luštrek, M.; Kaluža, B.; Dovgan, E.; Pogorelc, B.; Gams, M. Behavior analysis based on coordinates of body tags. In: *European Conference on Ambient Intelligence*, vol. 5859 of *Lecture Notes in Computer Science.* 14–23 (Springer, Salzburg, Austria, 2009).

Luštrek, M.; Nemec, B.; Kaluža, B.; Piltaver, R.; Gams, M.; Velez, I.; Larsson, K.; Gonzalez, N. *Report on Indoor Reconstruction and Interpretation Techniques.* Technical report (Confidence Project, Jozef Stefan Institute, Ljubljana, Slovenia, 2008).

Lymberopoulos, D.; Bamis, A.; Savvides, A. Extracting spatiotemporal human activity patterns in assisted living using a home sensor network. In: *Proceedings of the 1st International Conference on Pervasive Technologies Related to Assistive Environments*, PETRA '08. 29:1–29:8 (ACM, New York, NY, USA, 2008).

Mahajan, D.; Kwatra, N.; Jain, S.; Kalra, P.; Banerjee, S. A framework for activity recognition and detection of unusual activities. In: *Proceedings of the Fourth Indian Conference on Computer Vision, Graphics & Image Processing (ICVGIP).* 15–21 (Allied Publishers Private Limited, Kolkata, India, 2004).

Makris, D.; Ellis, T. Learning semantic scene models from observing activity in visual surveillance. *IEEE Transactions on Systems, Man and Cybernetics* **35**, 397–408 (2005).

Makris, D.; Ellis, T.; Black, J. Intelligent visual surveillance: Towards cognitive vision systems. *The Open Cybernetics & Systemics Journal* **2**, 219–229 (2008).

Markou, M.; Singh, S. Novelty detection: A review - part 1: Statistical approaches. *Signal Processing* **83**, 2481–2497 (2003).

Mirchevska, V.; Kaluža, B. Towards intelligent home caregiver. In: *Proceedings of the 1st Jožef Stefan International Postgraduate School Students' Conference.* 32–33 (IPS, Ljubljana, Slovenia, 2009).

Mirchevska, V.; Kaluža, B. Learning through interaction. In: *Proceedings of the 2nd Jožef Stefan International Postgraduate School Students' Conference.* 30 (IPS, Ljubljana, Slovenia, 2010).

Mirchevska, V.; Kaluža, B.; Luštrek, M.; Gams, M. Real-time alarm model adaptation based on user feedback. In: *Workshop on Ubiquitous Data Mining, European Conference on Artificial Intelligence (ECAI 2010).* 39–43 (IOS Press, Lisbon, Portugal, 2010).

Moeslund, T. B.; Granum, E. Pose estimation of a human arm using kinematic constraints. In: *Scandinavian Conference on Image Analysis.* 1–9 (IAPR, Bergen, Norway, 2001).

Monekosso, D.; Remagnino, P. Behavior analysis for assisted living. *IEEE Transactions on Automation Science and Engineering* **7**, 879–886 (2010).

Muncaster, J.; Ma, Y. Hierarchical model-based activity recognition with automatic low-level state discovery. *Journal of Multimedia* **5**, 66–67 (2007).

Murphy, K. P. Dynamic bayesian networks (unpublished book chapter, 2002).

Naftel, A.; Khalid, S. Classifying spatiotemporal object trajectories using unsupervised learning in the coefficient feature space. *Multimedia Systems* **12**, 227–238 (2006).

Navaratnam, R.; Thayananthan, A.; Torr, P. H. S.; Cipolla, R. Hierarchical part-based human body pose estimation. In: *Proceedings of British Machine Vision Conference.* 1–10 (BMVA Press, Oxford, United Kingdom, 2005).

Naylor, M.; Attwood, C. I. Advisor: Annotated digital video for intelligent surveillance and optimised retrieval. Final report (Thales Research & Technology Ltd, Berkshire, United Kingdom, 2003).

Nguyen, N. T.; Phung, D. Q.; Venkatesh, S.; Bui, H. Learning and detecting activities from movement trajectories using the hierarchical hidden markov models. In: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2005).* 955–960 (IEEE Computer Society Press, San Diego, CA, USA, 2005).

Niu, W.; Long, J.; Han, D.; Wang, Y.-f.; Barbara, S. Human activity detection and recognition for video surveillance. In: *IEEE International Conference on Multimedia and Expo.* 719–722 (IEEE Computer Society Press, Taipei, Taiwan, 2004).

Ohsawa, Y. *Chance Discoveries in Real World Decision Making: Data-based Interaction of Human Intelligence and Artificial Intelligence* (Springer, Berlin, Heidelberg, 2009).

Oliver, N.; Garg, A.; Horvitz, E. Layered representations for learning and inferring office activity from multiple sensory channels. *Computer Vision and Image Understanding* **96**, 163–180 (2004).

Oliver, N. M. *Towards Perceptual Intelligence: Statistical Modeling of Human Individual and Interactive Behaviors* (Ph. D. thesis, Massachusetts Institute of Technology, Cambridge, Massachusetts, United States, 2000).

Oliver, N. M.; Rosario, B.; Pentland, A. P. A bayesian computer vision system for modeling human interactions. *IEEE Transactions On Pattern Analysis And Machine Intelligence* **22**, 831–843 (2000).

OpenSteer. Steering behaviors for autonomous characters. http://opensteer.sourceforge.net (accessed: April 2011).

Park, K.; Lin, Y.; Metsis, V.; Le, Z.; Makedon, F. *Abnormal human behavioral pattern detection in assisted living environments* (ACM Press, New York, NY, USA, 2010).

Perš, J.; Kristan, M.; Perše, M.; Kovačič, S. Motion based human identification using histograms of optical flow. In: *Proceedings of the 12th Computer Vision Winter Workshop.* 19–26 (Institute for Computer Graphics and Vision, Graz University of Technology, Graz, Austria, 2007).

Piciarelli, C.; Micheloni, C.; Foresti, G. L.; Member, S. Trajectory-based anomalous event detection. *IEEE Transactions on Circuits and Systems for Video Technology* **18**, 1544–1554 (2008).

Piltaver, R.; Dovgan, E.; Gams, M. An intelligent indoor surveillance system. *Informatica* **35**, 383–390 (2011).

Protégé. Open source ontology editor and knowledge-base framework. http://protege.stanford.edu (accessed: November 2009).

Qian, G.; Guo, F.; Ingalls, T.; Olson, L.; James, J.; Rikakis, T. A gesture-driven multimodal interactive dance system. In: *IEEE International Conference on Multimedia and Expo.* 1579–1582 (IEEE Computer Society Press, Taipei, Taiwan, 2004).

Quah, J. T. S.; Sriganesh, M. Real-time credit card fraud detection using computational intelligence. *Expert Systems with Applications* **35**, 1721–1732 (2008).

Rabiner, L. R. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE* **77**, 257–286 (1989).

Regelous, S. MASSIVE: Multiple agent simulation system in virtual environment. http://www.massivesoftware.com (accessed: April 2011).

Rheinfurth, M.; Howell, L. W.; C., G. *Probability and statistics in aerospace engineering* (National Aeronautics and Space Administration, Marshall Space Flight Center, Springfield, VA, 1998).

Sanquist, T.; Sheridan, T.; Lee, J.; Cooke, N. Human factors aspects of anomaly detection systems. http://www.sis.pitt.edu/∼mlewis/surveillance/robocup10/COHSI_Anomaly_Dection.pdf (accessed: February 2013).

Schmidt, C. F.; Sridharan, N. S.; Goodson, J. L. The plan recognition problem: An intersection of psychology and artificial intelligence. *Artificial Intelligence* **11**, 45–83 (1978).

Sillito, R. R.; Fisher, R. B. Semi-supervised learning for anomalous trajectory detection. In: *Proceedings of British Machine Vision Conference.* 1035–1044 (British Machine Vision Association, Leeds, United Kingdom, 2008).

Stauffer, C.; Eric, W.; Grimson, W. E. L. Learning patterns of activity using real-time tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **22**, 747–757 (2000).

Steggles, P.; Gschwind, S. *The Ubisense smart space platform.* Technical report (Ubisense, Chesterton, UK, 2009).

Stephen, B.; Petropoulakis, L. An ambient software monitoring system for unsupervised user modelling. *Expert Systems with Applications* **28**, 557–567 (2005).

Storf, H.; Kleinberger, T.; Becker, M.; Schmitt, M.; Bomarius, F.; Prueckner, S. An event-driven approach to activity recognition in ambient assisted living. In: *Proceedings of the European Conference on Ambient Intelligence*, AmI '09. 123–132 (Springer, Berlin, Heidelberg, 2009).

Sugaya, M.; Ohno, Y.; van der Zee, A.; Nakajima, T. A lightweight anomaly detection system for information appliances. In: *International Symposium on Object/Component/Service-Oriented Real-Time Distributed Computing (ISORC).* 257–266 (IEEE Computer Society Press, Tokyo, Japan, 2009).

Sukthankar, G.; Sycara, K. A cost minimization approach to human behavior recognition. In: *Proceedings of the Fourth International Joint Conference on Autonomous Agents and Multiagent Systems*, AAMAS '05. 1067–1074 (ACM, New York, NY, USA, 2005).

Sukthankar, G.; Sycara, K. Hypothesis pruning and ranking for large plan recognition problems. In: *Proceedings of the Twenty-Third AAAI Conference on Artificial Intelligence.* 998–1003 (AAAI Press, Chicago, Illinois, USA, 2008).

Sun, T. H.; Tien, F. C. Using backpropagation neural network for face recognition with 2D + 3D hybrid information. *Expert Systems with Applications* **35**, 361–372 (2008).

Taleb, N. N. *The Black Swan* (Random House, New York, NY, USA, 2007).

Tambe, M.; Rosenbloom, P. S. RESC: An approach for dynamic, real-time agent tracking. In: *International Joint Conference on Artificial Intelligence (IJCAI 1995).* 499–522 (AAAI Press, Montreal, Quebec, Canada, 1995).

Tapia, E.; Intille, S.; Haskell, W.; Larson, K.; Wright, J.; King, A.; Friedman, R. Real-time recognition of physical activities and their intensities using wireless accelerometers and a heart rate monitor. In: *11th IEEE International Symposium on Wearable Computers.* 37–40 (IEEE Computer Society Press, Boston, MA, USA, 2007).

Tsai, J.; Kaminka, G.; Epstein, S.; Zilka, A.; Rika, I.; Wang, X.; Ogden, A.; Brown, M.; Fridman, N.; Taylor, M.; Bowring, E.; Marsella, S.; Tambe, M.; Sheel, A. ESCAPES - Evacuation simulation with children, authorities, parents, emotions, and social comparison. In: *International Conference on Autonomous Agents and Multiagent Systems.* 457–464 (IFAAMAS, Taipei, Taiwan, 2011).

Tung, F. *Goal-Based Trajectory Analysis for Unusual Behaviour Detection in Intelligent Surveillance* (Ph. D. thesis, University of Waterloo, Waterloo, ON, Canada, 2010).

Tušar, T.; Gams, M. Outlier detection in an access control system (in Slovene). In: *Proceedings of the 9th International Multiconference Information Society (IS 2006).* 136–139 (Jozef Stefan Institute, Ljubljana, Slovenia, 2006).

Vaswani, N.; Chowdhury, A. R.; Chellappa, R. Shape activity: A continuous state HMM for moving/deforming shapes with application to abnormal activity detection. *IEEE Transactions on Image Processing* **14**, 1603–1616 (2005).

Vicon. Motion Capture Systems from Vicon. http://www.vicon.com (accessed: November 2009).

Vilalta, R.; Drissi, Y. A perspective view and survey of meta-learning. *Artificial Intelligence Review* **18**, 77–95 (2002).

Visontai, M. Detecting unusual activity in video. In: *Proceedings of Computer Vision and Pattern Recognition 2004.* 819–826 (IEEE Computer Society, Washington, DC, USA, 2004).

Viterbi, A. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE Transactions on Information Theory* **13**, 260–269 (1967).

Waern, A.; Stenborg, O. *A simplistic approach to keyhole plan recognition.* Technical report (Swedish Institute of Computer Science, Kista, Sweden, 1995).

Wahyudi, W. A.; Syazilawati, M. Intelligent voice-based door access control system using adaptive-network-based fuzzy inference systems (ANFIS) for building security. *Journal of Computer Science* **3**, 274–280 (2007).

Wallace, S. Behavior bounding: An efficient method for high-level behavior comparison. *Journal of Artificial Intelligence Research* **34**, 165–208 (2009).

Wang, H. Intelligent agent-assisted decision support systems: Integration of knowledge discovery, knowledge analysis, and group decision support. *Expert Systems with Applications* **12**, 323–335 (1997).

Weiss, W. E. Dynamic security: An agent-based model for airport defense. In: *Proceedings of the 2008 Winter Simulation Conference.* 1320–1325 (WSC, Miami, Florida, USA, 2008).

Wilson, D. L. Intelligent video systems for perimeter and secured entry access control. In: *Proceedings of the 39th Annual IEEE International Carnahan Conference on Security Technology ICCST.* 260–262 (IEEE Computer Society Press, Las Palmas de G.C., Spain, 2006).

Witten, I. H.; Frank, E. *Data Mining: Practical Machine Learning Tools and Techniques, 2nd Edition* (Morgan Kaufmann, New York, NY, USA, 2005).

Wong, J.; Ho, S. Y. A local experts organization model with application to face emotion recognition. *Expert Systems with Applications* **36**, 804–819 (2009).

Woods, K.; Bowyer, K.; Jr, W. P. K. Combination of multiple classifiers using local accuracy estimates. In: *IEEE Conference on Computer Vision and Pattern Recognition.* 391 (IEEE Computer Society Press, Los Alamitos, CA, USA, 1996).

Yin, J.; Yang, Q.; Pan, J. J. Sensor-based abnormal human-activity detection. *IEEE Transactions on Knowledge and Data Engineering* **20**, 1082–1090 (2008).

Yin, L.; Yang, R.; Gabbouj, M.; Neuvo, Y. Weighted median filters: A tutorial. *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing* **43**, 157–192 (1996).

Zhang, C.; W. Guesgen, H.; Yeap, W. K.; Lühr, S.; Venkatesh, S.; West, G.; Bui, H. H. *PRICAI 2004: Trends in Artificial Intelligence*, vol. 3157 of *Lecture Notes in Computer Science* (Springer, Berlin, Heidelberg, 2004).

Zhang, Y.; Zhang, X. J.; Liu, Z. J. Irregular behavior recognition based on two types of treading tracks under particular scenes. In: *Proceedings of the Second International Conference on Knowledge Science, Engineering and Management (KSEM 2007).* 508–513 (Springer, Melbourne, Australia, 2007).

Appendices

Appendix A: Generating Suspicious Behavior

To simulate suspicious passenger behavior within ESCAPES simulator, we defined a new agent type going unnoticed from point A to point B as follows: suspicious agent's state contains the current position $Q_s(x, y)$. At each time step, the agent $\mathtt{s}$ computes the probability for being seen by any authority figure $\mathtt{a} \in \mathcal{S}$, where $\mathcal{S}$ is a set of authorities in a certain range. Similarly, a state of an authority agent $\mathtt{a}$ is defined by position $Q_a(x, y)$ and direction $\vec{d_a}$. Probability that the authority agents $\mathtt{a}$ sees another agent at distance r with an offset angle θ from the current direction $\vec{d_a}$ is defined as a bivariate normal distribution $N_a(r, \theta)$ as shown in Figure 12.1.

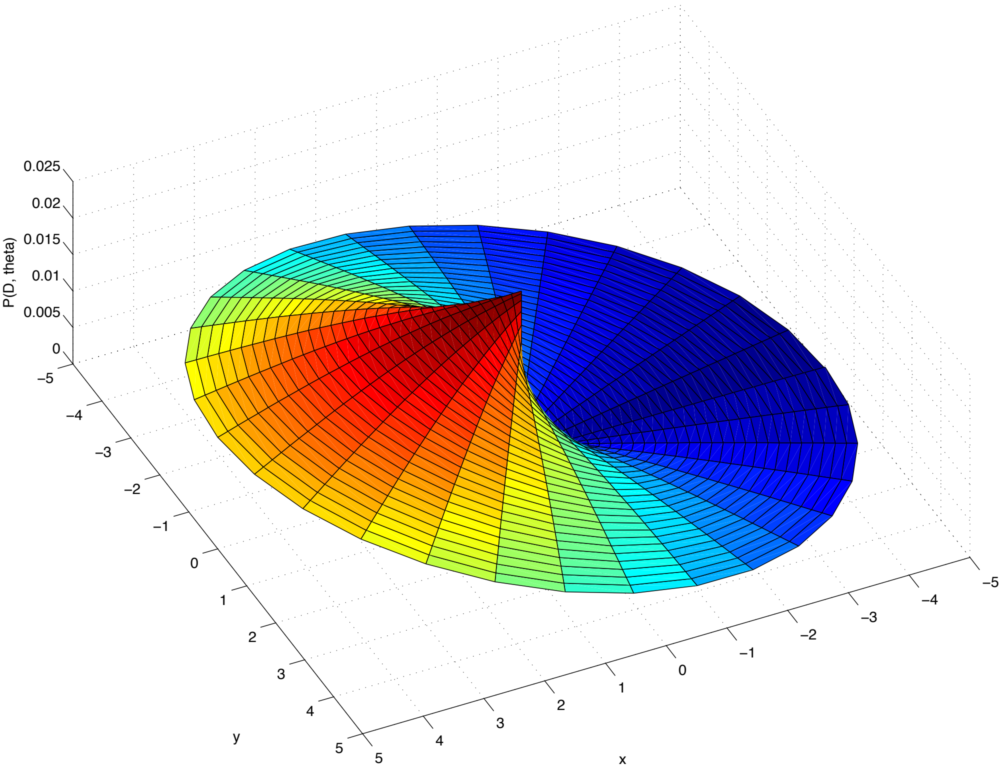


Figure 12.1: Authority's viewpoint modeled with bivariate normal distribution $N(r, \theta)$. Warmer color represents higher probability too see a particular point.

Points A and B are randomly chosen for each independent simulation. When the agent $\mathtt{s}$ reaches the point B, the simulation ends. The behavior of the suspicious agent follows a few simple rules:

1. Compute p as a sum of probabilities for being seen by any authority figure $\mathtt{a} \in A$ in the current position Q_s (and nearby $\pm\epsilon$ region)

$$p = \sum_{a \in A} \iint_{Q_s-\epsilon}^{Q_s+\epsilon} N_a(r,\theta) \tag{12.1}$$

2. If p exceeds a threshold value, then compute eight random points $c_i \in C$ in radius r, else restore the original final point B and go to step 4.

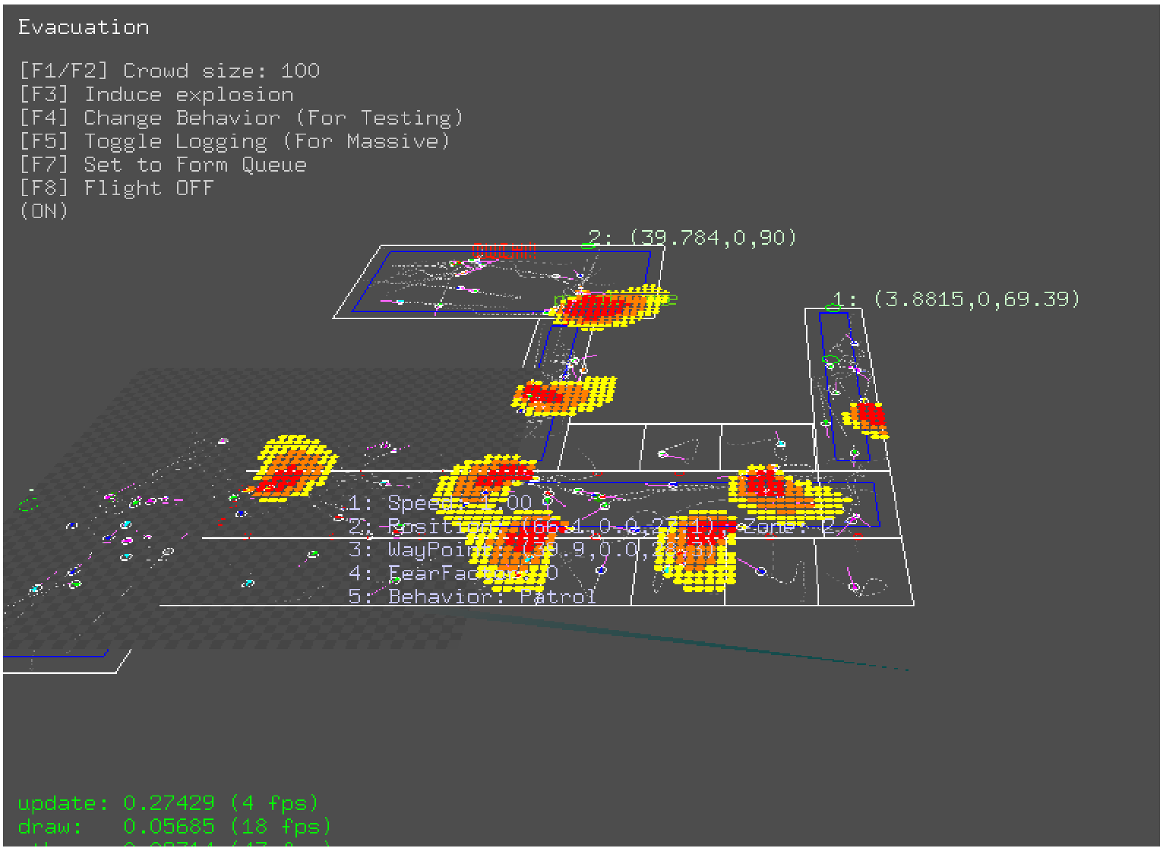
3. Select a point such that the sum of probabilities among the current point Q_s and the end point c_i is the smallest

$$\operatorname*{arg\,min}_{c_i \in C} \sum_{a \in A} \int_{Q_s}^{c_i} N_a(r,\theta)$$

   and define it as a new final point B'.

4. Move towards the final point. If the distance $d(Q_s, B) < \epsilon$, end, else go to step 1.

The resulting behavior is quite convincing and complex; ability to take into account several authorities and find the best solution in the given situation results in avoiding authorities in a half circle, making U-turns and continuing in the opposite direction, and even hiding in nearby stores. A visualization of the airport with viewpoint cones for eight authorities is shown in Figure 12.2.



```
Evacuation

[F1/F2] Crowd size: 100
[F3] Induce explosion
[F4] Change Behavior (For Testing)
[F5] Toggle Logging (For Massive)
[F7] Set to Form Queue
[F8] Flight OFF
(ON)
```

```
1: Speed: 1.00
2: Position: (66.1,0.0,27.1) Zone: 2
3: WayPoint: (39.9,0.0,28.3)
4: FearFactor: 0
5: Behavior: Patrol
```

```
update: 0.27429 (4 fps)
draw:   0.05685 (18 fps)
```

Figure 12.2: Authorities' viewpoints at the airport displayed with colormap: red for high and yellow for low probability.

Appendix B: Bibliography

B.1 Publications Related to This Thesis

B.1.1 Journal Articles

- Kaluža, B.; Cvetković, B.; Dovgan, E.; Gjoreski, H.; Mirchevska, V.; Gams, M.; Luštrek, M. Multiagent care system to support independent living. *International Journal on Artificial Intelligence Tools,* accepted (2013), **JCR: 0.22**

- Kaluža, B.; Gams, M. Analysis of daily-living dynamics. *Journal of Ambient Intelligence and Smart Enviroments* **4**, 403–413 (2012), **JCR: 0.63**

- Kaluža, B.; Dovgan, E.; Tušar, T.; Tambe, M.; Gams, M. A probabilistic risk analysis for multimodal entry control. *Expert Systems with Applications* **38**, 6696–6704 (2011c), **JCR: 2.203**

- Dovgan, E.; Kaluža, B.; Tušar, T.; Gams, M. Improving user verification by implementing an agent-based security system. *Journal of Ambient Intelligence and Smart Environments* **2**, 21–30 (2010b), **JCR: 1.5**

- Luštrek, M.; Kaluža, B. Fall detection and activity recognition with machine learning. *Informatica* **33**, 197–204 (2009)

B.1.2 Conference Papers

- Luštrek, M.; Kaluža, B.; Cvetković, B.; Dovgan, E.; Gjoreski, H.; Mirchevska, V.; Gams, M. Confidence: Ubiquitous care system to support independent living. In: *Proceedings of 20th biennial European Conference on Artificial Intelligence (ECAI 2012).* 1013–1014 (IOS Press, Montpellier, France, 2012).

- Kaluža, B.; Kaminka, G.; Tambe, M. Detection of suspicious behavior from a sparse set of multiagent interactions. In: *Eleventh International Conference on Autonomous Agents and Multiagent Systems, AAMAS 2012.* 955–964 (IFAAMAS, Valencia, Spain, 2012b).

- Kaluža, B.; Dovgan, E.; Luštrek, M.; Gams, M. Context aware MAS to support elderly people. In: *Eleventh International Conference on Autonomous Agents and Multiagent Systems, AAMAS 2012.* 1485–1488 (IFAAMAS, Valencia, Spain, 2012a).

- Dovgan, E.; Cvetković, B.; Mirchevska, V.; Kaluža, B.; Luštrek, M.; Gams, M. Improving the quality of life for elderly. In: *Confidence International Conference.* 99–108 (University of Jyväskylä, Jyväskylä, Finland, 2010a).

- Kaluža, B.; Mirchevska, V.; Dovgan, E.; Luštrek, M.; Gams, M. An agent-based approach to care in independent living. In: *International Joint Conference on Ambient Intelligence (AmI-2010)*, vol. 6439 of *Lecture Notes in Computer Science.* 177–186 (Springer, Malaga, Spain, 2010b).

- Kaluža, B.; Gams, M. An approach to analysis of daily living dynamics. In: *Proceedings of the World Congress on Engineering and Computer Science 2010*, vol. 1 of *International Conference on Machine Learning and Data Analysis, ICMLDA-10.* 485–490 (Newswood Limited, San Francisco, CA, USA, 2010).

- Luštrek, M.; Kaluža, B.; Dovgan, E.; Pogorelc, B.; Gams, M. Behavior analysis based on coordinates of body tags. In: *European Conference on Ambient Intelligence*, vol. 5859 of *Lecture Notes in Computer Science.* 14–23 (Springer, Salzburg, Austria, 2009).

- Kaluža, B.; Dovgan, E. Glajenje trajektorij gibanja človeškega telesa zajetih z radijsko tehnologijo. In: *Proceedings of the 12th International Multiconference Information Society*, vol. A. 97–100 (Jožef Stefan Institute, Ljubljana, Slovenia, 2009).

- Dovgan, E.; Kaluža, B.; Tušar, T.; Gams, M. Agent-based security system for user verification. In: *IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology.* 331–334 (IEEE Computer Society Press, Milan, Italy, 2009).

- Kaluža, B. Reducing spurious activity transitions in a sequence of movement. In: *Proceedings of the Eighteenth International Electrotechnical and Computer Science Conference*, vol. B. 163–166 (IEEE Slovenija, Portorož, Slovenia, 2009).

- Kaluža, B.; Luštrek, M. Fall detection and activity recognition methods for the Confidence project: A survey. In: *Proceedings of the 12th International Multiconference Information Society*, vol. A. 22–25 (Jožef Stefan Institute, Ljubljana, Slovenia, 2008).

- Kaluža, B.; Gams, M. Odkrivanje terorističnih vstopov s pomočjo ambientalne inteligence. In: *Proceedings of the Seventeenth International Electrotechnical and Computer Science Conference*, vol. B. 145–148 (IEEE Slovenija, Portorož, Slovenia, 2007a).

- Kaluža, B.; Gams, M. Intelligent access control system based on users behavior. In: *Procedings of the 2008 Networking and Electronic Commerce Research Conference.* 305–308 (ATSMA, Riva del Garda, Italy, 2008a).

B.1.2 Workshop Papers and Posters

- Kaluža, B.; Kozina, S.; Luštrek, M. Activity recognition repository: Towards competitive benchmarking in AmI. In: *Workshop on Activity Context Representation, AAAI Conference on Artificial Intelligence (AAAI-12).* 44–47 (AAAI Press, Toronto, Canada, 2012c).

- Kaluža, B.; Kaminka, G.; Tambe, M. Towards detection of suspicious behavior from multiple observations. In: *Workshop on Plan, Activity, and Intent Recognition, AAAI Conference on Artificial Intelligence (AAAI-11).* 8–18 (AAAI Press, San Francisco, California, 2011e).

- Kaluža, B.; Kaminka, G.; Tambe, M. Identifying suspicious behavior from multiple events. In: *3rd Jožef Stefan International Postgraduate School Students' Conference.* 74–79 (IPS, Ljubljana, Slovenia, 2011d).

- Kaluža, B.; Dovgan, E.; Cvetković, B.; Mirchevska, V.; Luštrek, M.; Gams, M. A multiagent system for remote eldercare. In: *Workshop on Agents for Ambient Assisted Living, International Conference on Practical Applications of Agents and Multi-Agent Systems.* 33–40 (Springer, Salamanca, Spain, 2011a).

- Kaluža, B.; Dovgan, E.; Mirchevska, V.; Luštrek, M.; Gams, M. Intelligent monitoring of the elderly in home environment. In: *AI and Health Communication, AAAI 2011 Spring Symposium Series.* 1 (AAAI Press, Stanford, California, USA, 2011b).

- Kaluža, B.; Mirchevska, V.; Dovgan, E. Context-aware MAS for remote elderly care. In: *Proceedings of the 2nd Jožef Stefan International Postgraduate School Students' Conference.* 26 (IPS, Ljubljana, Slovenia, 2010a).

- Kaluža, B.; Mirchevska, V.; Luštrek, M.; Vélez, I.; Gams, M. Ubiquitous care system to support independent living: Preliminary results. In: *Roots for the future of ambient intelligence: Adjunct proceedings.* 308–315 (Springer, Salzburg, Austria, 2009b).

- Kaluža, B.; Dovgan, E.; Tušar, T.; Gams, M. Intelligent risk analysis in access control. In: *Proceedings of the International Workshop on Quantitative Risk Analysis for Security Applications in conjunction with International Joint Conferences on Artificial Intelligence (IJCAI 2009)*, vol. A. 9–16 (AAAI Press, Pasadena, California, USA, 2009a).

- Mirchevska, V.; Kaluža, B. Towards intelligent home caregiver. In: *Proceedings of the 1st Jožef Stefan International Postgraduate School Students' Conference.* 32–33 (IPS, Ljubljana, Slovenia, 2009).

B.1.3 Popular Articles

- Gams, M.; Krivec, J.; Kaluža, B.; Rode, A. Varnostni sistem CIVaBiS: celoviti inteligentni varnostni biometrični sistem. *Obramba* **41**, 30–33 (2009).

- Kaluža, B.; Gams, M. Z ambientalno inteligenco nad teroriste. *Življenje in tehnika* **59**, 23–29 (2008b).

B.2 Other Publications

B.2.1 Conference Papers

- Cvetković, B.; Luštrek, M.; Kaluža, B.; Gams, M. Multi-classifier adaptive training: Specializing an activity recognition classifier using semi-supervised learning. In: *International Joint Conference on Ambient Intelligence*, vol. 7683 of *Lecture Notes in Computer Science.* 193–207 (Springer, Pisa, Italy, 2012).

- Gimon, D.; Gjoreski, H.; Kaluža, B.; Gams, M. Using accelerometers to improve position-based activity recognition. In: *Proceedings of the 13th International Multiconference Information Society (IS 2010).* 15–18 (Jožef Stefan Institute, Ljubljana, Slovenia, 2010).

- Cvetković, B.; Kaluža, B.; Gams, M. Prilagajanje modela za razpoznavanje aktivnosti človeka z aktivnim delno nadzorovanim učenjem. In: *Proceedings of the 13th International Multiconference Information Society.* 63–66 (Jožef Stefan Institute, Ljubljana, Slovenia, 2010).

- Gams, M.; Luštrek, M.; Kaluža, B. Patologija končno razložena? In: *Proceedings of the 11th International Multiconference Information Society*, vol. A. 19–21 (Jožef Stefan Institute, Ljubljana, Slovenia, 2008).

- Kaluža, B.; Gams, M. Razpoznavanje števk pri varnostnem geslu zvočne CAPTCHA zaščite. In: *Proceedings of the Seventeenth International Electrotechnical and Computer Science Conference*, vol. B. 371–372 (IEEE Slovenija, Portorož, Slovenia, 2007b).

- Kaluža, B.; Luštrek, M.; Gams, M. Patologija minimaksa v sintetičnih drevesih in Pearlovi igri. In: *Proceedings of the 10th International Multiconference Information Society*, vol. A. 84–87 (Jožef Stefan Institute, Ljubljana, Slovenia, 2007a).

- Kaluža, B.; Luštrek, M.; Gams, M.; Tavčar, A. Pathology in minimax searching. In: *Proceedings of the Sixteenth International Electrotechnical and Computer Science Conference*, vol. B. 107–110 (IEEE Slovenija, Portorož, Slovenia, 2007b).

- Gorup, Č.; Tavčar, A.; Kaluža, B. Eavesdroping on VoIP network. In: *Proceedings of the Sixteenth International Electrotechnical and Computer Science Conference*, vol. A. 69–72 (IEEE Slovenija, Portorož, Slovenia, 2007).

B.1.2 Workshop Papers and Posters

- Cvetković, B.; Luštrek, M.; Kaluža, B.; Gams, M. Semi-supervised learning for adaptation of human activity recognition classifier to the user. In: *Workshop on Space, Time and Ambient Intelligence, International Joint Conferences on Artificial Intelligence (IJCAI 2011).* 24–29 (AAAI Press, Barcelona, Spain, 2011).

- Mirchevska, V.; Kaluža, B. Learning through interaction. In: *Proceedings of the 2nd Jožef Stefan International Postgraduate School Students' Conference.* 30 (IPS, Ljubljana, Slovenia, 2010).

- Mirchevska, V.; Kaluža, B.; Luštrek, M.; Gams, M. Real-time alarm model adaptation based on user feedback. In: *Workshop on Ubiquitous Data Mining, European Conference on Artificial Intelligence (ECAI 2010).* 39–43 (IOS Press, Lisbon, Portugal, 2010).

Appendix C: Biography

Boštjan Kaluža received his Diploma in 2008 from the Faculty of Computer and Information Science, University of Ljubljana, Slovenia, by defending the thesis *Analysis of Minimax Pathology and Pearl Game*. He was awarded with the Prešeren Award (the highest faculty award) for his thesis.

In 2008, he enrolled in the *New Media and E-Science* program at the Jožef Stefan International Postgraduate School, Ljubljana, Slovenia. He held a young researcher scholarship for doctoral studies and research training awarded by the Slovene Research Agency.

He spent an academic year as a visiting researcher at the University of Southern California in the Teamcore research group under the advisement of Prof. Dr. Milind Tambe, where he studied anomalous and suspicious agent behavior.

Since October 2008, he has been working at the Department of Intelligent Systems at the Jožef Stefan Institute, under the advisement of Prof. Dr. Matjaž Gams. His research focuses on the development of novel algorithms and approaches, with an emphasis on analysis of agent behavior from sensor data. For his work, he has received two best student paper awards, one at the International Conference on Machine Learning and Data Analysis (San Francisco, 2010), and the other at the 3rd Jožef Stefan International Postgraduate School Students' Conference (Ljubljana, 2011).

Appendix D: Acknowledgments

First of all, I would like to thank my supervisor Prof. Dr. Matjaž Gams and co-supervisor Dr. Mitja Luštrek. They provided me guidance, support, understanding, and professional assistance of the most valuable kind. Their extensive discussions and insightful explorations have been of great value for this work, which could not have been carried out without them.

I am very grateful to my colleagues at the Department of Intelligent Systems at the Jožef Stefan Institute, several of whom deserve special merits. As regards the security domain, studied within the CIVaBiS project, I would like to thank the project team, in particular Tea Tušar, Erik Dovgan, Jana Krivec, Aleš Tavčar, Robert Blatnik, as well as the security experts from the Slovenian Ministry of Defense, the Špica International d.o.o. team, who provided the hardware components, and Dr. Janez Perš from the Faculty of Electrical Engineering at the University of Ljubljana for computer vision expertise.

The AAL domain experiments were made within the EU FP7 project Confidence. I am very thankful to the project team, in particular to Erik Dovgan, Violeta Mirchevska, Božidara Cvetković, Rok Piltaver, Barbara Tvrdi, Blaž Strle, and colleges from the Department for Automation, Biocybernetics and Robotics at the Jožef Stefan Institute, as well as anonymous volunteers that made experimental recordings possible.

I would like to express my deep gratitude to Prof. Dr. Milind Tambe for his encouragement, thoughtful guidance, support, and understanding during the course of research at the University of Southern California. I am also thankful to the members of the Teamcore research group for their useful discussions and friendly help. In particular, I would like to thank to Zhengyu Yin for valuable comments and suggestions regarding theoretical detection framework, Jason Tsai and Matthew Brown for their help with implementation challenges in the ESCAPES simulator, and James Pita, my office mate, for keeping me motivated. I am sincerely thankful to Prof. Gal Kaminka for constructive critisim and excellent insights. My warm thanks also go to Prof. Dr. Emma Bowring, Prof. Dr. Paul Scerri, Prof. Dr. Louis-Philippe Morency, and Prof. Dr. Ram Nevatia for providing valuable comments.

I am also very thankful to Prof. Dr. Bogdan Filipič and Prof. Dr. Marko Bohanec, who monitored my progress during the study and helped me to focus my research.

I wish to thank to my family, to my wife Ajda, sister Mateja, and parents Damjana and Janko. They believed in me and were always there when I needed them.

I also want to thank to my classmates and good friends Aleš Tavčar, Črt Gorup, Erik Dovgan, Dejan Petelin and Miha Mlakar, who helped me make my everyday commitments much easier, and Tony Guan, who made my Los Angeles experience exciting.

Last, but not the least, I am grateful to the Department of Intelligent Systems, Jožef Stefan Institute, and the Slovene Research Agency for providing me a scholarship, which made this thesis possible.

Index